Uma Plataforma de Serviços de Recomendação para Bibliotecas Digitais

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Daniel Carlos Guimarães Pedronette e aprovada pela Banca Examinadora.

Campinas, 28 de março de 2008.

Ricardo Torres

Prof. Dr. Ricardo Torres
Instituto de Computação - Unicamp
(Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Júlia Milani Rodrigues – CRB8a / 2116

Pedronette, Daniel Carlos Guimarães

P448p Uma plataforma de serviços de recomendação para bibliotecas digitais / Daniel Carlos Guimarães Pedronette -- Campinas, [S.P. :s.n.], 2008.

Orientador : Ricardo da Silva Torres

Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Bibliotecas digitais. 2. Sistemas de recomendação. 3. Serviços Web. I. Torres, Ricardo da Silva. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

(mjmr/imecc)

Título em inglês: A plataform of recommendation services for digital libraries.

Palavras-chave em inglês (Keywords): 1. Digital libraries. 2. Recommendation systems. 3. Web services.

Área de concentração: Sistemas de Informação

Titulação: Mestre em Ciência da Computação

Banca examinadora:

Prof. Dr. Ricardo da Silva Torres (IC-UNICAMP)
Prof. Dr. Renato Fileto (UFSC)
Profa. Dra. Islene Calciolari Garcia (IC-UNICAMP)
Profa. Dra. Ariadne Maria Brito Rizzoni Carvalho (IC-UNICAMP)

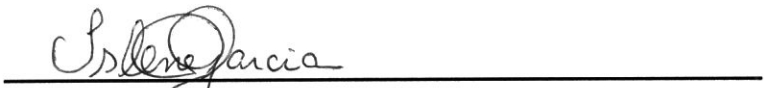
Data da defesa: 28/03/2008

Programa de Pós-Graduação: Mestrado em Ciência da Computação

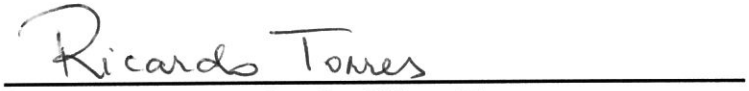
TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 28 de março de 2008, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Renato Fileto
DIE – UFSC



Profª. Drª. Islene Calciolari Garcia
IC – UNICAMP



Prof. Dr. Ricardo da Silva Torres
IC – UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Uma Plataforma de Serviços de Recomendação para Bibliotecas Digitais

Daniel Carlos Guimarães Pedronette

Março de 2008

Banca Examinadora:

- Prof. Dr. Ricardo Torres
  Instituto de Computação - Unicamp (Orientador)
- Prof. Dr. Renato Fileto
  Universidade Federal de Santa Catarina
- Profª. Drª. Islene Calciolari Garcia
  Instituto de Computação - Unicamp
- Profª. Drª. Ariadne Maria Brito Rizzoni Carvalho (Suplente)
  Instituto de Computação - Unicamp

Resumo

Em virtude do crescimento acelerado de conteúdo nas mais diversas aplicações de bibliotecas digitais, a tarefa de localizar objetos digitais de interesse é cada vez mais desafiadora. Sob essa perspectiva, técnicas de recomendação procuram prover, de acordo com as preferências do usuário final, alternativas de escolha de objetos mantidos em uma biblioteca digital.

Essa dissertação concentra-se em aspectos relacionados às técnicas de recomendação e suas interações com aplicações de bibliotecas digitais. Uma plataforma de serviços de recomendação, chamada RecS-DL, é proposta, visando ampliar as possibilidades de utilização das ferramentas de recomendação. A Plataforma RecS-DL apresentada é independente de domínio de aplicação, de tecnologias e técnicas de recomendação. O serviço de recomendação oferecido pode ser facilmente agregado a bibliotecas digitais clientes, assim como novos mecanismos de recomendação podem ser acoplados à plataforma de maneira dinâmica. Este trabalho também apresenta uma especificação formal da plataforma de serviços de recomendação proposta a partir do Arcabouço 5S. Para isso foram propostas novas definições e extensões de conceitos deste arcabouço.

Por fim, são apresentados os resultados obtidos a partir de testes realizados com a plataforma. Experimentos foram conduzidos considerando bibliotecas digitais reais e avaliações por potenciais usuários. Resultados experimentais ratificam a hipótese de que a plataforma facilita a interoperabilidade de ferramentas de recomendação em bibliotecas digitais.

Abstract

The increasing amount of data in the most diverse digital libraries applications makes the process of finding relevant digital objects a challenging task. From this perspective, recommendation techniques can provide, according to user preferences, relevant digital objects stored in a digital library.

This dissertation focuses on recommendation techniques and their interactions with digital libraries applications. A platform for recommendation services, called RecS-DL, has been proposed to support the use of recommendation tools. The proposed RecS-DL Platform is independent of application domain, technology, and recommendation techniques. The recommendation services offered by the platform can be easily incorporated into digital libraries systems. Furthermore, new recommendation engines can also be plugged into the platform in a dynamic way. This work also presents a formal specification of the proposed platform, using the 5S Framework. To do this, new definitions and extensions of this framework are proposed.

Finally, we present the results obtained from tests performed with the platform. Experiments were conducted considering real digital libraries and evaluations made by potential users. Experimental results confirm that the platform facilitates the interoperability of recommendation tools in digital libraries systems.

Agradecimentos

Esta dissertação representa um dos frutos de mais uma fase de aprendizados. Frutos de uma árvore que simboliza mais do que um trabalho, mas também a minha formação, pessoal e acadêmica, e para a qual tantos dispensaram atenção e cuidados.

O meu objetivo nas linhas seguintes é expressar a minha gratidão e o meu reconhecimento de que, sem o apoio dessas pessoas, esse fruto não teria sido possível, ou estaria mais azedo ou mais amargo. Tarefa difícil, dada a contribuição de tantas pessoas. Então, caso alguma injustiça venha a ser cometida, aproveito para deixar de antemão o meu pedido de desculpas, reforçado pela honestidade das intenções.

Agradeço sinceramente ao meu orientador Prof. Dr. Ricardo, por vários motivos. Por ter sido um real orientador, acompanhando e direcionando os trabalhos em seus menores passos. Pelos ensinamentos, nos aspectos mais diversos, desde gramática a método científico. Por ter-me permitido considerá-lo como amigo, dispensando aos seus alunos atenção de inestimável valor humano.

Agradeço aos meus pais Carlos e Marilena, a minha esposa Daniele, ao Guilherme e ao meu filho Giovanni. A todos pelo amor incondicional que tonifica a vida a cada dia. Aos meus pais por tudo que sempre me deram: amor, lições e bons hábitos. Só a medida que evelhecemos conseguimos perceber o quanto deles levamos conosco. A Dani, pelo nosso muito grande amor, que dá luz e ânimo para enfrentar todas as dificuldades e desafios. Pela atenção e pelo cuidado de todo dia. Ao Guilherme por todo o carinho e todos os questionamentos. Ao Giovanni, pelo simples fato de fazer parte da minha vida, despertando, ao simples observar das suas maneiras de criança, o que de melhor há em mim.

Agradeço a minha família, de forma geral. Aos meus tios e tias, meus primos e primas, que não são muitos, mas muitas vezes merecem lugares especiais em nossas vidas. Em especial aos meus avós Raul e Genir, que estarão presentes em meu coração, a cada momento da minha vida.

Agradeço aos meus amigos. Independente de quão próximos podemos estar hoje, levo um pouco de todos eles comigo. Cronologicamente, desde a infância: Paulo, Leandro, Bruno, Paulinho, Cristiano. Do Colégio Técnico: Éderson, Gabi, Tobias, Melina, Davi,

Daniel. Da Unicamp: Maurício Calixto, Marcelo, Ronaldo, Luiz Carlos. Da faculdade: Djalma, Cláudio, Delano. Do carro: Alexandre, Danilo. E tantos outros.

Agradeço a todos os meus professores. De todas as épocas da minha vida. Estou certo que cada um deles pôde contribuir, um pouco que seja, para que eu pudesse terminar esse trabalho.

Agradeço aos colaboradores da Unicamp, às pessoas e às Insituições que representam. Agradeço ao Queiroz pelas contribuições em relação à Biblioteca Digital da Unicamp, e pelo apoio sempre pronto e irrestrito. Agradeço à Marilda (CCUEC) e ao Marcos Dario (DGA) pela possibilidade de cursar as disciplinas do curso.

Agradeço aos colaboradores da Biblioteca Central da Unicamp, em especial ao Vicentini pela colaboração e atenção dispensada. Agradeço aos alunos de pós-graduação do IC, da turma de Bibliotecas Digitais do Prof. Ricardo, que participaram dos experimentos.

Agradeço aos colaboradores do Instituto de Computação. À Comissão de Pós-Graduação do IC e aos projetos da FAPESP, FAEPEX, CAPES e CNPq pelo apoio financeiro a minha viagem ao SBBD para apresentação desse trabalho. Agradeço às professoras Prof. Dr^{a}Islene e Prof. Dr^{a}Ariadne que contribuíram, desde a qualificação, para alguns direcionamentos desse trabalho.

> "A principal meta da educação é criar homens que sejam capazes de fazer coisas novas, não simplesmente repetir o que outras gerações já fizeram. Homens que sejam criadores, inventores, descobridores. A segunda meta da educação é formar mentes que estejam em condições de criticar, verificar e não aceitar tudo que a elas se propõe." (Jean Piaget)

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Uma biblioteca digital pode ser definida, de maneira bastante ampla, como uma coleção organizada de objetos, mecanismos para navegação e busca em ambientes distribuídos e um conjunto de serviços com o objetivo de atender às necessidades de usuários [23]. Assim, bibliotecas digitais são sistemas de informação avançados e complexos, indo muito além das tradicionais máquinas de busca, dado que oferecem um amplo conjunto de serviços voltados para comunidades de usuários específicas [62]. Há anos as aplicações de bibliotecas digitais apresentam um crescimento vertiginoso, seja em quantidade de informações, seja em abrangência de domínios de utilização. Processo análogo ocorre nas aplicações de *e-commerce* e diversas outras aplicações que tem a *Web* como plataforma.

Em decorrência desse crescimento, entretanto, a localização e escolha do conteúdo desejado têm se tornado um grande desafio. Diversas metodologias vêm sendo desenvolvidas com o objetivo de auxiliar o usuário: métodos eficientes de busca [35], métodos que exploram o relacionamento semântico de informações [8], uso de metadados [18], técnicas de recomendação [31, 71], entre outros.

O processo de recomendação, já bastante difundido em relações sociais humanas, tem um importante papel também nos sistemas de bibliotecas digitais [58]. Atualmente, os sistemas de recomendação estão presentes nos mais diversos segmentos, tais como recomendação de músicas [12], vídeos [9], livros [31], entre outros. Este projeto de pesquisa concentra-se em aspectos relacionados às técnicas de recomendação e suas interações com aplicações de bibliotecas digitais.

O principal objetivo das técnicas de recomendação consiste em auxiliar os usuários nas tarefas de escolha e localização de conteúdo. Essas técnicas dão aos sistemas a funcionalidade de agir de maneira pró-ativa, apresentando sugestões para o usuário. Funcionalidades como essas são de grande valia quando o usuário possui pouco conhecimento sobre determinada área, ou mesmo para especialistas, sobretudo quando as bases de dados gerenciadas têm crescimento acelerado.

Segundo Gonçalves [23], recomendar consiste em: *dada uma coleção, um ator e um conjunto de avaliações para os objetos dessa coleção (definidas pelo mesmo, ou por outros atores) produzir um subconjunto da coleção para esse ator particular.*

Sob tal definição, as recomendações podem ser dadas de forma direta por atores humanos, provendo avaliações sobre um determinado objeto conhecido. Estas informações, por sua vez, serão utilizadas para produzir as recomendações para outros indivíduos. Outras técnicas operam de forma mais autônoma, sem intervenção direta aparente dos usuários. Algumas delas utilizam informações de conteúdo dos objetos [47], outras fazem uso das informações de histórico dos usuários para inferir relações entre atores e objetos [31].

Uma taxonomia bastante utilizada na literatura [31, 58, 71] divide as técnicas em dois grandes grupos: filtragem baseada em conteúdo (*content-based filtering*) e filtragem colaborativa (*collaborative-filtering*). Como não são mutuamente exclusivas, algumas técnicas combinam características de ambas, dando origem a abordagens híbridas.

As técnicas de *Collaborative Filtering* operam recomendando objetos para atores, baseando-se no histórico de satisfação de outros usuários semelhantes. De maneira geral, as recomendações são geradas em decorrência da observação do comportamento dos usuários no sistema. Perfis de usuários são criados a partir do registro de interações anteriores - histórico de comportamento - e funções de similaridade são utilizados para encontrar usuários semelhantes.

As técnicas de *Content-Based Filtering* recomendam objetos que são semelhantes quanto ao conteúdo a outros objetos aos quais o ator já demonstrou interesse. Embora as buscas sejam realizadas a partir de atributos dos objetos, a relação entre ator e objetos também é obtida a partir do histórico de interação do ator com o sistema.

Ambas as técnicas apresentam deficiências [71]. Técnicas baseadas em conteúdo são de difícil utilização em domínios não textuais (imagens, vídeos) e limitam-se à recomendação de objetos com conteúdo semelhante. Técnicas baseadas em filtragem colaborativa são pouco efetivas quando há uma relação desproporcional entre muitos objetos e poucos atores. Sob esse contexto, surgem as técnicas híbridas, procurando unir o melhor das abordagens [32, 65]. Assim, o maior desafio nesses casos consiste em definir como fazer com que as técnicas operem de forma conjunta e produtiva.

Alvo de pesquisas desde a década de 90 [22], as técnicas de recomedação são funcionalidades já bastante difundidas e utilizadas, dado que há diversas ferramentas de recomendação descritas na literatura [14, 42, 49] e em utilização em diversas aplicações de mercado [2, 20]. Dessa forma, além da eficácia das técnicas, características relacionadas à infra-estrutura das ferramentas são bastante relevantes no contexto dos sistemas de recomendação. Vários aspectos estruturais dessas ferramentas geralmente limitam suas possibilidades de aplicação e reuso. Entre esses aspectos podem-se destacar:

- **Domínio de aplicação:** as ferramentas são projetadas tendo em vista uma bi-

blioteca digital ou domínio específico, impossibilitando sua reutilização em outros cenários;

- **Técnicas de recomendação:** os pacotes disponíveis são, em sua maioria, sistemas monolíticos. Os serviços oferecidos não são extensíveis e geralmente baseiam-se em uma única técnica de recomendação;

- **Tecnologia:** a implementação das ferramentas é realizada em linguagem, plataforma ou interface específica, limitando o acesso de aplicações clientes.

O presente trabalho apresenta como principal objeto de pesquisa a especificaçã e implementação da Plataforma RecS-DL. A plataforma de serviços de recomendação proposta apresenta soluções aos problemas citados, como independência de técnica de recomendação, tecnologias empregadas e domínio de aplicação. A metodologia aplicada consistiu em um levantamento inicial das ferramentas de recomendação existentes. Em seguida, analisaram-se as características das ferramentas de recomendação identificadas. Essa análise, por sua vez, produziu um conjunto de especificações decorrentes dos requisitos de flexibilidades desejados.

Baseado nesse conjunto de requisitos e nos problemas apresentados pelas ferramentas analisadas, foram apresentadas soluções que constituem a base da plataforma proposta. A utilização de metadados e a escolha de termos genéricos para designar as informações armazenadas apresentaram-se como soluções para a independência de domínio de aplicação. O desenvolvimento de *engines* de recomendação, semelhante ao conceito de *plugins*, possibilitou a independência quanto a técnicas de recomendação. Dessa forma, várias técnicas de recomendação podem ser oferecidas pela plataforma. A utilização de serviços web possibilitou a utilização da plataforma por aplicações clientes implementadas em linguagens e tecnologias diversificadas.

A definição e a especificação formal dos serviços e *interfaces* oferecidos consistiu em outro desafio de pesquisa. Nesse sentido, foram analisadas metodologias de especificação e arcabouços formais [23] que poderiam ser utilizados no projeto e implementação de bibliotecas digitais, seus serviços e componentes. O Arcabouço 5S [23] foi selecionado e utilizado para definição e extensão de conceitos relacionados a *software* e recomendação. Baseando-se nas definições criadas e na consolidação das especificações e serviços oferecidos, a plataforma proposta foi formalmente definida.

Diante das considerações acima, a contribuição central deste trabalho consiste na especificação, modelagem e implementação de uma plataforma de serviços de recomendação para bibliotecas digitais. Essa plataforma tem como principal objetivo reduzir as limitações das ferramentas existentes, ampliando as possibilidades de utilização de serviços de recomendação em bibliotecas digitais. Entre os requisitos observados na plataforma

proposta, destacam-se a flexibilidade sob diversos aspectos, propiciando a independência de domínio de aplicação, de técnicas de recomendação e tecnologias utilizadas.

A plataforma proposta foi testada inicialmente utilizando um *dataset* de avaliação de filmes e um sub-conjunto da Biblioteca Digital da Unicamp [73]. Posteriormente foi avaliada por potenciais usuários e integrada a outras bibliotecas digitais. Resultados experimentais demonstram que a plataforma pode efetivamente oferecer serviços de recomendação para bibliotecas digitais, contribuindo para a interoperabilidade de ferramentas de recomendação.

O restante deste documento é organizado como segue:

- o Capítulo 2 apresenta os trabalhos correlatos;
- o Capítulo 3 discute o modelo arquitetural e aspectos de implementação da plataforma proposta;
- o Capítulo 4 estende conceitos de recomendação e define formalmente a plataforma a partir do Arcabouço 5S;
- o Capítulo 5 apresenta os experimentos realizados para validação da plataforma criada;
- por fim, o Capítulo 6 apresenta as conclusões e considerações finais desta dissertação, assim como possíveis trabalhos futuros.
- os Apêndices contém as definições das estruturas de Banco de Dados da Plataforma RecS-DL e os questionários utilizados nos experimentos com potenciais usuários.

Capítulo 2

Trabalhos relacionados

Este capítulo tem como objetivo apresentar os trabalhos relacionados aos conceitos utilizados pela plataforma proposta. Serão discutidos aspectos gerais das técnicas de recomendação, suas principais classificações e respectivas características. Serão abordados aspectos de infra-estrtura das ferramentas de recomendação existentes, vantagens e desvantagens das abordagens apresentadas. Será discutida também a tecnologia de Serviços Web, que foi utilizada na implementação da plataforma. Por fim, será apresentado o Arcabouço 5S [23], seus principais conceitos e características. O Arcabouço 5S será utilizado para definir formalmente a plataforma proposta.

2.1 Técnicas de recomendação

É frequente a necessidade de fazer escolhas e localizar itens de interesse dentre um grande conjunto possibilidades. No dia-a-dia, recorre-se às recomendações de outras pessoas para essa tarefa. Os sistemas de recomendação, por sua vez, procuram auxiliar de forma automática nesse processo [61]. A recomendação como processo social representa um importante papel em diversos segmentos, sejam comerciais ou científicos, já que é extremamente custoso para um determinado indivíduo conhecer todas as possíveis alternativas de escolha em um dado domínio [30].

Um sistema de recomendação é um sistema que recomenda um conjunto de itens presente em uma grande coleção de itens disponíveis, para um determinado usuário de acordo com os seus interesses. Um item pode ser qualquer objeto digital que o usuário tenha interesse, como um livro, filme ou artigo [32]. Os interesses dos usuários podem ser extraídos a partir de origens diversas, como histórico de comportamento, perfil do usuário, entre outros. Dessa forma, os sistemas de recomendação procuram auxiliar o usuário nos processos de escolha e localização de conteúdo.

O processo de recomendação opera basicamente a partir de três conceitos:

- **Dados de Histórico:** informações que o sistema dispõe antes do início do processo de recomendação;
- **Dados de Entrada:** informação fornecida no momento do pedido de recomendação;
- **Técnica de Recomendação:** combina os dados de entrada e de histórico produzindo recomendações.

No momento em que uma recomendação é solicitada, um dado de entrada deve ser informado. Esse dado de entrada pode ser um ator ou um objeto da biblioteca digital. Os dados de entrada são processados por técnicas de recomendação e combinados com dados de histórico. Os dados de histórico, por sua vez, podem ser de várias origens, como histórico de comportamento, perfil de usuário, ou índices de semelhança entre objetos. O resultado da técnica de recomendação é formado por um subconjunto da biblioteca digital que consiste no conjunto de recomendações geradas. A Figura 2.1 ilustra esse processo.

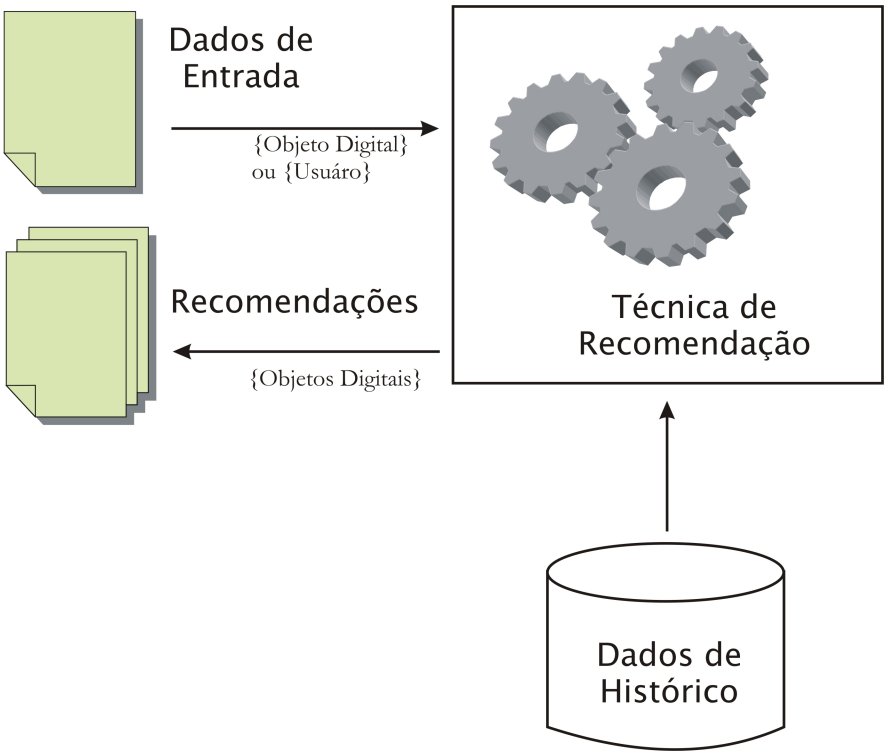


Figura 2.1: Visão geral do processo de recomendação.

As técnicas de recomendação existentes podem ser divididas em duas grandes categorias: filtragem colaborativa (*collaborative filtering*) e filtragem baseada no conteúdo

(*content-based filtering*). O termo filtragem colaborativa foi inicialmente cunhado visando designar um tipo de sistema específico no qual a filtragem de informação era realizada com o auxílio humano, ou seja, pela colaboração entre os grupos de interessados [58]. As técnicas baseadas em conteúdo geram recomendações a partir de informações textuais, palavras-chaves e técnicas de recuperação de informações. Mais recentemente, estudos têm sido realizados sobre metodologias para combinar as duas abordagens, dando origem a diversos trabalhos com técnicas híbridas [32, 65, 71]. A relação entre indivíduos e a formação de redes sociais também têm sido alvo de estudos [45, 53]. A Tabela 2.1 mostra uma breve comparação entre os dois tipos de técnicas de recomendação.

Tabela 2.1: Comparação de Técnicas de Recomendação.

Técnica	Vantagens	Desvantagens
Filtragem Colaborativa	Pode identificar nichos *cross*-gênero; Conhecimento de domínio não é necessário; Adaptativo: qualidade melhor com o tempo; *Feedback* implícito é suficiente.	Problema do novo usuário; Problema do novo item; Problema da ovelha negra; Depende de grande conjunto de dados de histórico;
Baseada em Conteúdo	Conhecimento de domínio não é necessário; Adaptativo: qualidade melhor com o tempo; *Feedback* implícito é suficiente.	Problema do novo usuário; Depende de conjunto de dados de histórico;

As duas técnicas apresentam várias vantagens em comum: ambas não necessitam coletar explicitamente informações do usuário, dado que apenas o *feedback* implícito é suficiente (histórico de comportamento, por exemplo). Ambas as técnicas melhoram seu desempenho com o tempo e quantidade de treinamento e não necessitam de conhecimento de domínio específio para operar. Uma vantagem exclusiva da filtragem colaborativa é a possibilidade de identificar recomendações, mesmo que não haja similaridade entre os itens recomendados, identificando nichos *cross*-gênero.

O problema do novo item, apresentado pela filtragem colaborativa, ocorre devido ao fato de que, quando da inclusão de um item em uma base de dados, o sistema de recomendação não possui nenhuma informação a respeito desse novo item. Dessa forma, não é possível que o item seja eficientemente recomendado para um usuário até que avaliações ou informações implícitas sejam obtidas através de outro usuário.

O problema do novo usuário, que afeta ambas as técnicas, consiste na ausência de informações sobre o usuário, impossibilitando qualquer tipo de recomendação segundo suas preferências.

Dado um usuário que apresente preferências não comuns, o sistema de recomendação

terá dificuldades para encontrar outros usuários com preferências similares. Suas recomendações podem se tornar de má qualidade, configurando o "*problema da ovelha negra*", ou problema da similaridade [58].

Embora ambas as abordagens baseiem-se nos dados de histórico de comportamento, esse problema é mais latente na abordagem colaborativa. Na verdade, a principal limitação de utilidade da abordagem colaborativa consiste no problema da esparsidade, que se refere à situação em que dados transacionais ou de *feedback* são esparsos ou insuficientes para idenficar preferências do usuário [30]. Isso ocorre, por exemplo, quando o número de itens é desproporcionalmente maior que o número de usuários.

Diante das vantagens e desvantagens apontadas, os sistemas de recomendação têm utilizado numerosas variações e combinações dessas técnicas, de acordo com o objetivo e domínio de aplicação. Alguns trabalhos correlatos foram analisados e suas principais características são apresentadas na Tabela 2.2. Como é possível verificar na tabela, apesar das variações de abordagens algorítimicas, sistemas de recomedação em diversos domínios de aplicação baseiam-se nas técnicas colaborativas, baseadas em conteúdo ou na combinação destas. Na próxima seção serão descritas com maiores detalhes as principais técnicas utilizadas pelos sistemas de recomendação, assim como algumas aplicações em trabalhos correlatos.

Vários termos relacionados à recomendação são utilizados neste trabalho. Haja visto que tais termos podem ter significados semelhantes, será brevemente discutida a conotação em que é utilizada cada um deles. O termo *sistemas de recomendação* designa, de forma mais geral, todos os sitemas que incluem *funcionalidades/serviços de recomendação*. As *técnicas de recomendação* referem-se a maneira como são implementados os serviços de recomendação. As *ferramentas de recomendação* são técnicas de recomendação implementadas em pacotes de *software*, que podem ser utilizadas em vários sistemas. Por fim, o termo *infra-estrutura* é utilizado para designar as tecnologias e técnicas utilizadas nas ferramentas.

2.1.1 Filtragem colaborativa

As técnicas de filtragem colaborativa (*collaborative filtering*), também chamadas de *social filtering* [67], baseiam-se principalmente no compartilhamento de experiências entre indivíduos que apresentam interesses semelhantes. Assim, as ações e análises de um usuário, considerando uma porção particular de informação, é armazenada para benefício de toda uma comunidade [29].

As experiências podem ser registradas por meio de avaliações, histórico de comportamento, histórico de compras ou de buscas, entre outros. O conteúdo da informação influencia na forma como os dados são obtidos: uma avaliação de determinado item é

Tabela 2.2: Trabalhos em sistemas de recomendação e técnicas utilizadas

Trabalho	Domínio de aplicação	Técnicas utilizadas	Abordagens Algorítmicas	Avaliação e validação
Recommendations for a video DL [9]	Vídeos	*Content-based Filtering*	TF-IDF [63] e *Spreading activation.*	Comparação a métodos de *collaborative filtering.*
Learning to Advertise [37]	Anúncios	*Content-based Filtering*	TF-IDF e Programação Genética.	Conjunto de dados de 100 páginas e 1.860 anúncios.
PocketLens [44]	Portável	*Collaborative Filtering*	Variantes de algoritmos colaborativos *item-to-item.*	Comunidades de 2.000 usuários.
TechLens [71]	Artigos científicos	Híbridas	Algoritmos colaborativos *user-based* e TF-IDF.	*Off-line*, com 100 mil artigos. *On-Line*, com 110 usuários.
Graph recommender system [31]	Livros	Híbridas	Grafos e Redes Neurais.	*Off-Line*, com 100 consumidores. Análise qualitativa por dois usuários.

explicitamente dada pelo usuário; informações de perfil são obtidas implicitamente a partir do comportamento do usuário no sistema. Essas formas levam a uma subclassificação desses sistemas segundo à automatização da coleta de informações.

Sistemas colaborativos não automáticos requerem que o usuário determine as relações com a comunidade, exigindo uma ampla carga coginitiva deste usuário. Como resultado, esses sistemas apresentam aplicações em comunidades pequenas e fechadas, onde os interesses de outros usuários sejam conhecidos. Em contrapartida, os sistemas colaborativos automáticos tornam transparente o processo de coleta de informações dos usuários, suportando recomendações para grandes comunidades de usuários anônimos [29].

O Tapestry [22], primeiro sistema criado segundo esta abordagem, visa prover um filtro de mensagens eletrônicas, baseado nas avaliações de outros usuários. O sistema permite ao usuário especificar uma consulta como: "mostre-me todos os memorandos que uma determinada pessoa considera importante". Assim, membros de determinada comunidade podem ser beneficiados pela experiência de outros [58].

O sistema GroupLens [36] foi a primeira proposta de filtragem colaborativa automática. O sistema utiliza-se de avaliações de usuários para determinadas notícias, numa escala de 1 a 5, para prover recomendações e predições aos demais vizinhos.

A abordagem de filtragem colaborativa tem sido utilizada nos mais diversos domínios. Aplicações têm sido propostas para recomendações de vídeos [9, 26], músicas [12], livros [31], artigos científicos [15], entre outros. Os sistemas de bibliotecas digitais, de uma forma geral, têm incorporado essa abordagen nos serviços de recomendação.

Nas abordagens de filtragem colaborativa, atores são tipicamente representados a partir dos objetos com os quais interagiu. Em uma biblioteca digital de 10.000 teses, por exemplo, cada ator é representado por um vetor de 10.000 posições, onde cada posição do vetor refere-se a um determinado objeto. A informação armazenada em cada posição do vetor pode ser um domínio (1 a 5, por exemplo), representando a nota dada para aquele objeto, ou uma valor binário, contendo 1 se o usuário ativo já interagiu com aquele objeto, ou 0, caso contrário. A Tabela 2.3 apresenta um exemplo desse tipo de estrutura de dados. A primeira linha da tabela por exemplo indica que o *Ator 1* atribuiu nota 4 ao *Objeto 2*, nota 3 ao *Objeto 3* e nota 5 ao *Objeto 4*.

Tabela 2.3: Matriz de Informações para filtragem colaborativa.

	Objeto 1	Objeto 2	Objeto 3	Objeto 4
Ator 1		4	3	5
Ator 2	3	2	4	
Ator 3	1	5		1
Ator 4		3	4	

As informações de relacionamento entre atores-objetos são comumentemente representadas na forma matricial. Na literatura, entretanto, há várias abordagens que apresentam essas informações em grafos bipartidos [30, 31, 45, 46]. Em [31], a atividade de recomendação é proposta como tarefa de busca em grafos. Em [46] é apresentado um *framework* para estudo de algoritmos em grafos em termos de "pulos" (*jumps*), que relacionam pessoas a aterfatos. A Figura 2.2 ilustra a representação destes relacionamentos: um grafo bipartido onde os vértices representam Atores ou Objetos e as arestas a relação entre um Ator e um Objeto. Ao lado é ilustrada uma matriz de adjacência, estrutura de dados comumente utilizada para armazenar esse tipo de informação.

Dentre os algoritmos mais utilizados nas técnicas de filtragem colaborativa, predominam os métodos baseados na "vizinhança". Segundo [58], a técnica *k-nearest-neighbor* (k vizinhos mais próximos) pode ser descrita em três fases:

1. Cálculo da similaridade de cada usuário em relação ao usuário para o qual deseja-se gerar a recomendação. A similaridade calculada a partir de uma métrica de similaridade refere-se à similaridade entre as preferências dos usuários em questão. Diversas métricas de similaridade são apresentadas com mais detalhes em [29], e a mais comumentemente utilizada é correlação de Pearson [60].

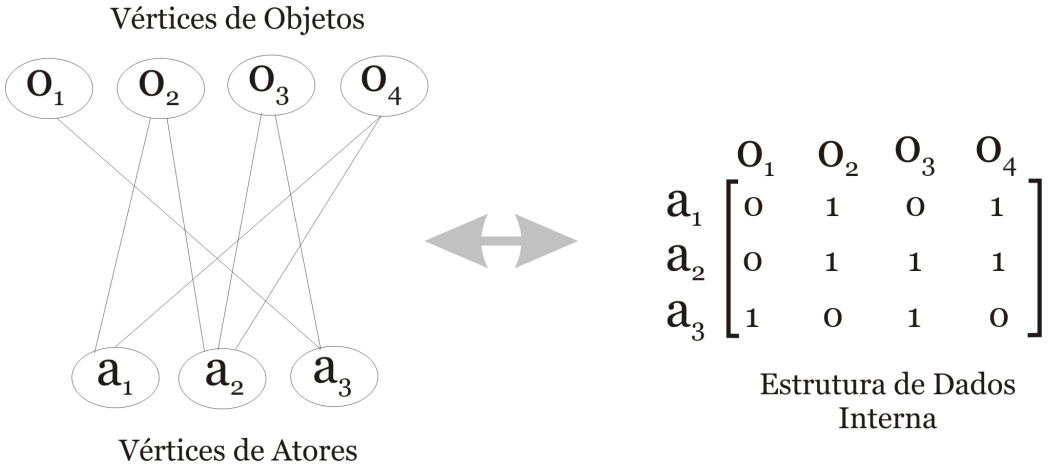


Figura 2.2: Representação em grafos de informações de relacionamento atores-objetos.

2. Selecionar um subconjunto de usuários com maiores similaridades (vizinhos) para considerar na predição.

3. Normalizar as avaliações e computar as predições ponderando as avaliações dos vizinhos com seus pesos.

Diversas outras técnicas também são aplicáveis aos sistemas de filtragem colaborativa. São apresentadas na literatura abordagens estatísticas [75,76], utilizando redes neurais [31] e ontologias [41].

2.1.2 Filtragem baseada em conteúdo

A filtragem baseada em conteúdo (*content-based filtering*) é caracterizada pela recomendação de objetos, baseando-se na correlação entre o conteúdo de itens e preferências dos usuários em relação a estes itens.

Uma forma possível de implementação da filtragem baseada em conteúdo consiste em utilizar avaliações de itens realizadas pelos usuários, que indicam se são ou não de seu interesse. Uma vez realizada a avaliação, o sistema busca itens que se assemelham em conteúdo com aqueles que foram classificados como de interesse, desconsiderando os demais [58].

Sistemas deste tipo apresentam limitações como: o conteúdo de dados não textuais é difícil de ser analisado (imagens, vídeos); o entendimento do conteúdo do texto pode ser prejudicado devido ao uso de sinônimos; pode ocorrer um processo de especialização demasiada, já que o sistema procura se basear em avaliações feitas pelo usuário. Outra

grande dificuldade na construção de técnicas baseadas em conteúdo consiste em extrair características do conteúdo que sejam realmente indicativas e significativas [75].

Entretanto as técnicas baseadas em conteúdo podem ser aplicadas com sucesso em domínios textuais, apresentando duas vantagens [32]: não apresenta o *"problema do primeiro item"* e não apresenta o problema de esparsidade.

A primeira vantagem baseia-se no fato de que uma recomendação ocorre quando um item e o perfil do usuário compartilham palavras em comum. Dessa forma, não é estritamente necessário que um item tenha sido avaliado antes que possa ser recomendado. A segunda é devido ao fato de que, em domínios textuais, as recomendações podem ser computadas de acordo com a similaridade entre o texto do item e o perfil do usuário, o que não depende de grande histórico de avaliações.

2.1.3 Algoritmos híbridos

Os algoritmos de filtragem híbrida consistem na combinação de técnicas de filtragem colaborativa e filtragem baseada em conteúdo, visando aliar as vantagens e reduzir as desvantagens de cada técnica.

Observando em maiores detalhes as características de ambas as técnicas, filtragem colaborativa e baseada em conteúdo, vemos que são complementares sob alguns aspectos. Por exemplo, a abordagem por conteúdo não apresenta o problema do novo item, nem o problema de esparsidade de dados. Em contrapartida a filtragem colaborativa não apresenta os problemas de especialização e dependência de domínio de conteúdo [32].

Dado que todas as técnicas conhecidas têm vantagens e desvantagens, uma linha de pesquisa comum em sistemas de recomendação converge para métodos e algoritmos que sejam capazes de combinar as recomendações de técnicas diferentes. Em [11], é apresentada uma taxonomia para esses métodos, assim como exemplos de sistemas que utilizam cada um deles:

- *Weighted*: as pontuações obtidas de diferentes técnicas são ponderadas e combinadas visando produzir uma única recomendação;
- *Switching*: dependendo da situação o sistema escolhe a técnica a ser utilizada;
- *Mixed*: recomendações obtidas de diferentes técnicas são apresentadas simultaneamente;
- *Feature Combination*: características de diferentes fontes de recomendações são implementadas em conjunto, em um único algoritmo de recomendação;
- *Cascade*: uma técnica refina as recomendações dadas por outra;

- *Feature Augmentation*: os resultados de uma técnica são utilizados como entrada para outras;
- *Meta-level*: o modelo aprendido por uma técnica de recomendação é usada como entrada para outra.

Recentemente, as técnicas híbridas têm sido alvo de recorrentes estudos no sentido de aumentar a eficácia dos sistemas de recomendação. Em [31, 66, 71, 75] são encontradas variações de aplicações de técnicas híbridas em sistemas de recomendação.

2.2 Infra-estrutura das ferramentas de recomendação

A difusão das divervas técnicas de recomendação originou outros desafios de pesquisa: como proporcionar que estes algoritmos sejam utilizados de forma comum por diversas aplicações clientes e em tecnologias e domínios diferentes; como torná-los configuráveis a cada aplicação e acessíveis por meio de *interfaces* padronizadas. Esta dissertação apresenta a seguir uma análise das abordagens de ferramentas atuais e propõe em seguida algumas contribuições para solucionar os problemas de infra-estrutura existentes.

Alguns pacotes e ferramentas têm sido propostos visando prover acesso a algoritmos de recomendação. As ferramentas CoFE - Colaborative Filtering Engine [14] e Multilens [43], por exemplo, podem ser utilizadas e acopladas às aplicações que necessitem de técnicas de recomendação. São independentes de domínio de aplicação e suportam armazenamento e configuração de banco de dados relacionais. Possibilitam a extensão de mecanismos de recomendação, todavia, baseando-se apenas no modelo colaborativo. Não permitem a utilização de outras técnicas de recomendação e têm limitações quanto à *interface* de acesso, baseada exclusivamente na tecnologia Java, o que impossibilita sua utilização por aplicações construídas em outras linguagens.

O pacote Duine [72] consiste em uma ferramenta que suporta várias técnicas de recomendação e configuração de parâmetros para os algoritmos disponíveis. A ferramenta também é independente de domínio e possibilita o armazenamento das informações em banco de dados. Todavia, a ferramenta apresenta limitações relacionadas ao uso de linguagem específica pelas aplicações clientes. Também há limitações quanto às bases de dados, já que só foram realizados testes na ferramenta utilizando um único SGBD. Já a abordagem apresentada nesta dissertação apresenta soluções que a tornam independente, tanto de técnicas de recomendações, quanto da linguagem da aplicação cliente ou SGBDs utilizados.

O projeto EasyUtil [19] oferece um serviço de recomendação colaborativo operando sobre o protocolo HTTP. As requisições são realizadas por meio de parâmetros enviados a uma determinada URL e os resultados são codificados em formato XML. A ferramenta

suporta aplicações em linguagens e domínios diversos. Entretanto, apresenta desvantagens, dado que se limita às técnicas colaborativas e não oferecem interfaces formalmente definidas, baseando-se apenas na URL do serviço.

O pacote Taste [49] consiste em uma ferramenta de recomendação baseada no modelo colaborativo. A ferramenta pode ser acoplada a aplicações clientes ou pode ser acessada a partir de interface de Serviços Web. Embora portável tecnologicamente, a ferramenta limita-se às técnicas colaborativas de recomendação. O projeto *The MobLife Recommender* [55] também oferece um serviço de recomendação baseado em Serviços Web e interfaces formalmente definidas em WSDL [13]. O algoritmo de recomendação utilizado é baseado em técnicas colaborativas é o *Tree Augmented Naive Bayesian Classifiers* (TAN). A utilização de Serviços Web possibilita boa portabilidade, todavia, a ferramenta também não permite a utilização de outras técnicas de recomendação. A Seção 2.3 apresenta as principais características da tecnologia de Serviços Web.

Embora a literatura a respeito dos sistemas de recomendação seja bastante vasta, nem sempre os sistemas tornam-se ferramentas disponíveis para livre utilização. A Tabela 2.4 apresenta um resumo de algumas dessas ferramentas previamente discutidas e algumas das suas características. Pode-se verificar na tabela que as ferramentas variam quanto à independência de técnica de recomendação e tecnologia de acesso. Há ferramentas que são independentes de técnicas de recomendação, mas vinculadas a linguagem de implementação específica. Em contrapartida há ferramentas independente de tecnologia, mas restritas a uma técnica de recomendação.

Tabela 2.4: Infra-estrutura das ferramentas de recomendação.

Ferramenta	Técnica	Acesso
Multilens	Colaborativa (extensível)	Java
CoFE	Colaborativa (extensível)	Java
Duine	Várias	Java
EasyUtil	Colaborativa	HTTP
Taste	Colaborativa	Serviços Web
The MobLife Recommender	Colaborativa	Serviços Web
Plataforma RecS-DL	**Várias**	**Serviços Web**

A Plataforma RecS-DL, proposta nesta dissertação e discutida em detalhes no Capítulo 3, unifica as vantagens de cada solução, integrando Serviços Web e flexibilidade quanto ao uso de técnicas de técnicas de recomendação. Assim, a plataforma torna-se independente de linguagem de implementação e domínio das aplicações clientes, assim como das técnicas de recomendação utilizadas.

2.3 Serviços Web

Segundo definição do consórcio W3C [10], um Serviço Web é um sistema de software projetado para prover interoperabilidade em interações entre máquinas através de uma rede. Possui interfaces descritas em formatos passíveis de processamento, especificamente WSDL [13], e interagem com outros sistemas por meio de mensagens SOAP [27], usando tipicamente protocolos HTTP.

Os Serviços Web estão se tornando padrão de desenvolvimento de aplicações distribuídas, permitindo que sistemas cooperem facilmente e compartilhem informações e funcionalidades. Espera-se que esta evolução altere a forma como as aplicações são construídas e desenvolvidas, como a informação é apresentada e compartilhada, e como software é adquirido.

A possibilidade de composição de serviços representa uma importante vantagem dos Serviços Web. A composição visa prover um modelo eficiente, baseado na Arquitetura Orientada a Serviços (SOA - *Service-Oriented Architecture*), no qual novos serviços podem ser criados e adicionados facilmente a aplicações existentes em tempo real [39].

Diferentes padrões e linguagens têm sido propostos para a composição de Serviços Web [33]. Entre outros, a linguagem BPEL [70] (*Business Process Execution Language por Web Services*) é um dos padrões emergentes para descrição do comportamento dos serviços. Diversas outras abordagens são encontradas na literatura: em [40], é proposta a composição de serviços baseadas em agentes de software; em [21, 64] são apresentadas abordagens para composições dinâmicas e semi-automáticas de serviços.

Uma arquitetura orientada a serviços difere de sistemas orientados a objetos e procedurais principalmente na forma de ligação entre os componentes (*binding*), já que são agentes de softwares autônomos, executados em ambientes heterogêneos. Sistemas tradicionais estabelecem relações entre os elementos baseando-se em tipos, nomes e requerendo implementações comuns. Serviços interagem baseando-se nos contratos e esquemas determinados pelas funções oferecidas. Isto permite que o serviço descreva a estrutura de suas mensagens, caracterizando suas interfaces e dispensando um ambiente compartilhado de execução.

A arquitetura de Serviços Web utiliza diversas tecnologias e padrões inter-relacionados. A Figura 2.3, retirada de [10], ilustra alguns dos padrões e tecnologias envolvidos e o relacionamento entre eles. O padão WSDL descreve o Serviço Web, especificando o formato das mensagens SOAP. O padrão UDDI está relacionado ao processo de descoberta de Serviços Web, funcionando como um serviço de busca de descrições WSDL. O XML e tecnologias diretamente relacionadas, como DTD e Schema representam papel fundamental nessa arquitetura, atuando como tecnologias base para as demais e garantindo requisistos de interoperabilidade.

Outro conceito importante consiste no fato de que as mensagens enviadas e recebidas pelos Serviços Web têm de trafegar através de algum meio de comunição. Vários protocolos podem ser utilizados com o objetivo de prover esse meio, como FTP, SMTP, JMS, sendo que o mais comumente utilizado é o HTTP. Outras características de infraestrutura e segurança podem ser agregadas às mensagens como criptografia, autenticação e gerenciamento de tráfego.

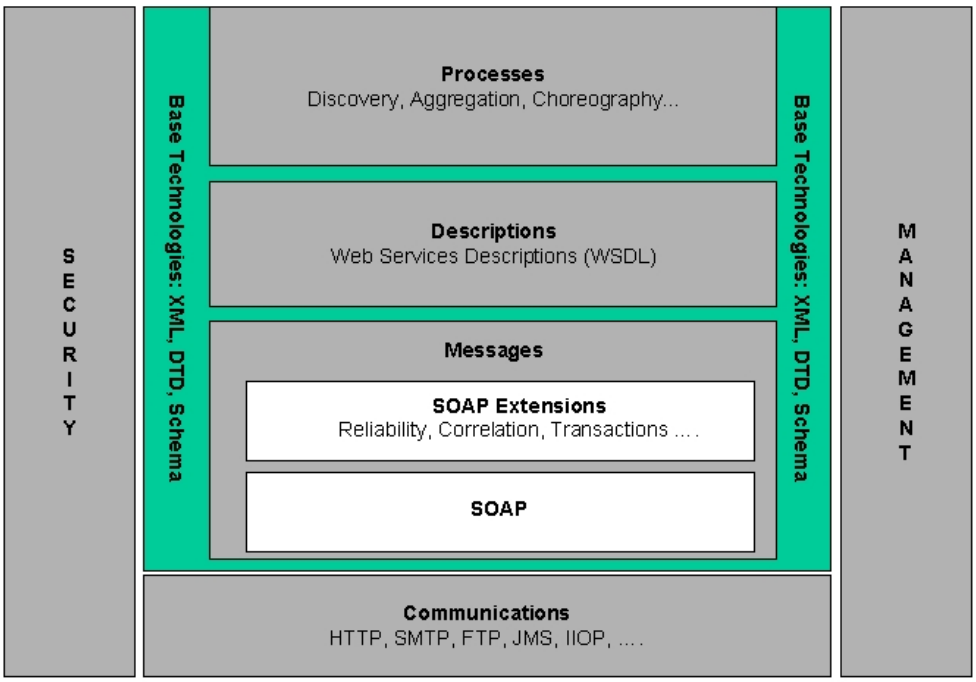


Figura 2.3: Arquitetura de Serviços Web (retirada de [10]).

No contexto de bibliotecas digitais, a interoperabilidade entre sistemas também têm sido alvo de diversas pesquisas. Em [5, 54] são apresentadas aplicações de Serviços Web visando constituir arquiteturas e serviços para bibliotecas digitais por meio de interfaces SOAP. Em [68] é proposta uma arquitetura para bibliotecas digitais, constituída pela composição de Serviços Web.

2.4 Arcabouço 5S

Devido à complexidade dos sistemas de bibliotecas digitais, em geral não há consenso sobre as definições e terminologias relacionadas a estes sistemas. Diferentes visões (históricas, tecnológicas) e perspectivas (biblioteconomia, recuperação de informações, interação humano-computador) levaram a diferentes definições. Isto se deve, entre outros fatos, à interdisciplinaridade inerente a esse tipo de sistema, integrando gerenciamento de banco de dados, interação homem-computador, recuperação de informações e serviços multimídias [23].

A necessidade de acomodar todas essas características complexas torna difícil o processo de compreensão e especificação de conceitos e funcionalidades desses sistemas. Assim, também a construção de novos sistemas de bibliotecas digitais torna-se um processo bastante custoso. Sob essa perspectiva, novas teorias e arcabouços vêm sendo desenvolvidos, visando entender e especificar as interações que ocorrem nos sistemas de bibliotecas digitais.

Teorias e arcabouços formais são cruciais para especificar e entender de forma clara e não ambígua as características, estruturas e o comportamento de sistemas de informação complexos, como os sistemas de bibliotecas digitais. Arcabouços formais abstraem as características gerais e comuns de um conjunto de sistemas desenvolvidos para problema similares, explicando suas estruturas, processos e práticas comuns. Além disso, podem ser utilizados no projeto de sistemas reais, provendo especificações precisas de requisitos. Estas especificações, por sua vez podem ser comparadas à implementação realizada, visando verificações de corretude [23].

O Arcabouço 5S [23] consiste em um arcabouço formal para especificação e modelagem de bibliotecas digitais. Esta seção apresenta brevemente esse arcabouço, que será posteriormente utilizado para especificação da Plataforma RecS-DL.

O Arcabouço 5S provê formalismos, uma linguagem própria e uma unificação tanto teórica quanto prática de componentes, interações e serviços relacionados a bibliotecas digitais. Este arcabouço baseia-se nas definições de:

- *Streams*: tipos de informações multimídia suportados pela biblioteca digital;
- *Structures*: como as informações são estruturadas e organizadas;
- *Spaces*: propriedades e operações dos componentes da biblioteca digital;
- *Scenarios*: serviços e comportamento da biblioteca digital;
- *Societies*: conjunto de atores e gerenciadores que atuam nos serviços.

O Arcabouço 5S define formalmente todos os principais conceitos relacionados a bibliotecas digitais. Um mapa das principais definições e seus relacionamentos é ilustrada na Figura 2.4, adaptada de [23]. A figura ilustra os conceitos e sua utilização, representada pelas setas, para a composição de conceitos mais complexos, até a definição de uma biblioteca digital mínima. A figura ilustra também a importância das várias perspectivas (*Streams, Structures, Spaces, Scenarios, Societies*) na definição de conceios mais complexos. Uma visão mais detalhada dessas perspectivas, em termos de primitivas, formalismos e objetivos é apresentada na Tabela 2.5.

Tabela 2.5: Visão geral do Arcabouço 5S [24].

Modelos	Primitivas	Formalismos	Objetivos
Stream	Texto, vídeo, áudio, imagens, softwares.	Sequências, tipos.	Descreve propriedades de conteúdo da BD, como codificação e linguagem para material textual ou formas particulares de dados multimídia.
Structures	Coleção, catálogo, hipertexto, documento, metadados, ferramentas de organização.	Grafos, nós, *links*, rótulos.	Especifica aspectos organizacionais do conteúdo da BD.
Spaces	Interface de usuário, índices, modelos de recuperação.	Conjuntos, operações, espaços, espaços vetoriais, de mensuração, de probabilidades.	Define visões lógicas e de apresentação de diversos componentes da BD.
Scenarios	Serviço, evento, condição, ação.	Diagramas de sequência, diagramas de colaboração.	Detalhes do comportamento dos serviços.
Societies	Comunidades, gerenciadores, atores, relacionamentos, atributos, operações.	Modelagem orientada a objetos. Padrões de projeto.	Define responsáveis pela execução serviços, atores que usarão determinados serviços, e o relacionamento entre eles.

O Arcabouço 5S permite ainda extensões de conceitos abordados e a descrição formal de sistemas de biblitecas digitais. O Capítulo 4 tem como objetivo estender algumas definições do Arcabouço 5S e utilizá-lo para definir formalmente a plataforma proposta nesta dissertação.

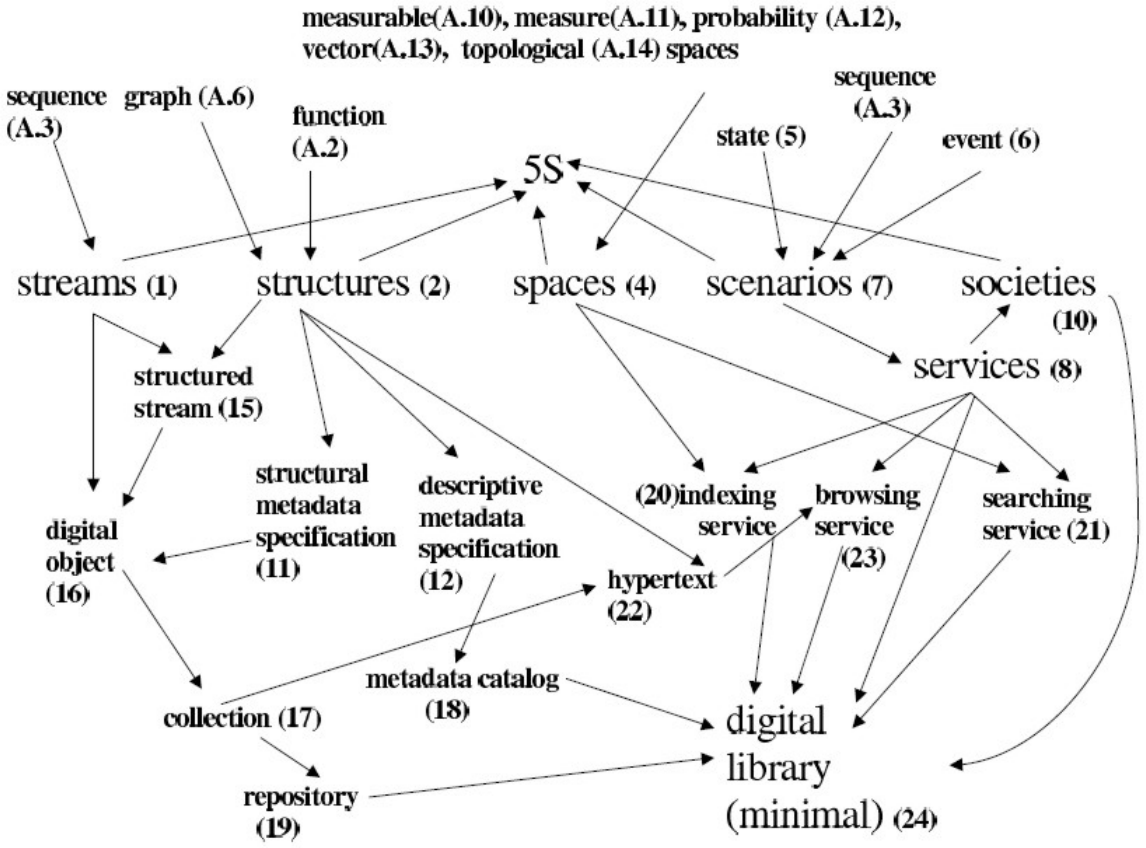


Figura 2.4: Mapa das definições formais do Arcabouço 5S.

Capítulo 3

Plataforma RecS-DL

Este capítulo tem como principal objetivo apresentar a Plataforma RecS-DL. Serão discutidos aspectos relacionados aos desafios de pesquisas enfrentados, à arquitetura da plataforma e à sua implementação.

3.1 Descrição da abordagem proposta

As implementações de técnicas de recomedação oferecem serviços já bastante difundidos e utilizados, havendo diversas ferramentas de recomendação descritas na literatura e em utilização em diversas aplicações de mercado. Entretanto, vários aspectos estruturais dessas ferramentas geralmente limitam suas possibilidades de aplicação e reuso. Entre esses aspectos podem-se destacar:

- **Domínio de aplicação:** as ferramentas são projetadas tendo em vista uma Biblioteca Digital ou domínio específico, impossibilitando sua reutilização em outros cenários;
- **Técnicas de recomendação:** os pacotes disponíveis são, em sua maioria, sistemas monolíticos. Os serviços oferecidos não são extensíveis e geralmente baseiam-se em uma única técnica de recomendação;
- **Tecnologia:** a implementação das ferramentas é realizada em linguagem, plataforma ou interface específica, limitando o acesso de aplicações clientes.

A abordagem proposta consiste em criar uma plataforma flexível o bastante para tornar-se independente de domínio de aplicação, de tecnologias utilizadas e extensível sob o ponto de vista de técnicas de recomendação. Assim, serão discutidas a seguir soluções para cada um dos problemas apresentados.

A solução da abordagem proposta em face à independência de domínio de aplicação consistiu na utilização de termos genéricos e abrangentes para designar as informações armazenadas. Assim um *item* da biblioteca digital pode ser um livro, uma tese ou um filme. O conteúdo do item, por sua vez, pode ser a tese completa, um resumo do livro, a sinopse ou vídeo do filme. Dessa forma, a *interface* definida pela plataforma pode ser utilizada da mesma maneira por bibliotecas digitais de domínios diversos. Essa abordagem foi implementada por meio da utilização do padrão Dublin Core [17] na modelagem dos repositórios de dados e metadados mantidos pela Plataforma RecS-DL. A plataforma permite também o armazenamento de dados binários, possibilitando a operação de diferentes tipos de arquivos além dos formatos textuais, como imagens, áudios, entre outros. Maiores detalhes da modelagem de dados da plataforma serão discutidos na Seção 3.4.1.

Outro requisito importante consiste na extensibilidade da plataforma, possibilitando a utilização de técnicas de recomendação diversas. A solução proposta consiste em uma estrutura de *engines*, semelhante ao conceito de *plugins*. Os *engines* podem ser desenvolvidos e instalados de maneira uniforme na plataforma. Dada uma evolução ou surgimento de uma técnica de recomendação, a *interface* comum permanece e pode continuar a ser utilizada da mesma maneira para um novo *engine*. Isso possibilita que desenvolvedores trabalhem de forma paralela, criando ou aperfeiçoando técnicas e algoritmos de recomendação e aplicando-as sobre interfaces comuns providas pela Plataforma RecS-DL. As estruturas criadas para os *engines* de recomendação serão discutidos na Seção 3.4.2.

Por fim, sob o ponto de vista tecnológico, foram selecionadas tecnologias abrangentes e padrões já bem aceitos e difundidos em aplicações acadêmicas e comerciais. A plataforma utilizou Serviços Web para suas *interfaces*, que tornaram os serviços de recomendação portáveis e acessíveis entre ambientes tecnológicos distintos. Essa abordagem permitiu que a plataforma possa ser utilizada por *softwares* implementadosem linguagens e sistemas operacionais diferentes daqueles utilizados pela plataforma. Foi considerada também a independência em relação a servidores de aplicação e banco de dados, de forma que a plataforma possa operar de forma transparente entre servidores distintos.

3.2 Desafios de Pesquisa

A definição da arquitetura da plataforma foi realizada tendo como principal objetivo o atendimento dos requisitos de flexibilidade propostos. O atendimento desses requisitos, por sua vez, resultou em diversos desafios de pesquisa. Os principais desafios enfrentados são listados a seguir:

- **Quais as funcionalidades que deveriam ser oferecidas às bibliotecas digitais clientes?**

A definição das funcionalidades de recomendação desejadas para a plataforma demandou uma análise das ferramentas e técnicas existentes. O fruto desse trabalho permitiu a definição das *interfaces* da Plataforma RecS-DL, discutidas e especificadas formalmente no Capítulo 4.

- **Como possibilitar a utilização de várias técnicas e ferramentas de recomendação utilizando uma mesma *interface* de acesso?**

  A solução proposta consistiu na definição de interfaces comuns e na criação do conceito de *engines* de recomendação. Dessa forma, a plataforma oferece acesso comum às funcionalidades de recomendação possibilitando ainda que ferramentas existentes sejam encapsuladas em *engines*.

- **Qual a estrutura dos *engines* de recomendação?**

  Para que o conceito de *engine* (*plugin*) proposto se tornasse factível, foi necessário criar um padrão para a construção desses *engines*. Essa padrão é basicamente composto por diretrizes para implementação em tecnologia Java e na definição de um conjunto de metadados para descrição do *engine*.

- **Quais serviços deveriam ser oferecidos pela plataforma aos *engines* de recomendação?**

  Para que os *engines* de recomendação pudessem executar as técnicas de recomendação era necessário o acesso às informações das bibliotecas digitais clientes. Dessa forma, uma API (*Application Programming Interface*) deveria ser definida e oferecida aos *engines* para execução dos acessos necessários.

- **Como viabilizar o acesso da plataforma às informações das bibliotecas digitais clientes?**

  A Plataforma RecS-DL necessita acessar as informações das bibliotecas digitais clientes como dados de entrada para execução das técnicas de recomendação. Dada essa necessidade foi definido o Módulo de Aquisição da plataforma, discutido em detalhes na Seção 3.4.1.

- **Como garantir flexibilidade para a criação e configuração de bibliotecas digitais na plataforma?**

  A plataforma pode ser utilizada por várias bibliotecas digitais clientes que podem apresentar características e configurações bastante distintas. De modo a garantir a flexibilidade da plataforma, foi criado um Serviço de Configuração, que será apresentado em detalhes na Seção 3.5.

- **Como efetuar uma modelagem de dados que atendesse às necessidades das diversas técnicas de recomendação?**

  O objetivo de independência de técnica de recomendação trouxe a necessidade de armazenamento de um grande conjunto de informações, capazes de abranger todas, ou pelo menos a maioria das técnicas. Assim, são armazenadas tanto informações de conteúdo dos itens (para técnicas baseadas no conteúdo) como avaliações de itens (para técnicas colaborativas). Além disso, é possível que um engine de recomendação crie estruturas de dados auxiliares.

3.3 Arquitetura da Plataforma RecS-DL

Abordando os desafios citados, foi criado um modelo arquitetural extensível e cujas tecnologias foram selecionadas segundo critérios de portabilidade. A Figura 3.1 ilustra a arquitetura geral da plataforma sob diversos aspectos, discutidos ao decorrer deste capítulo.

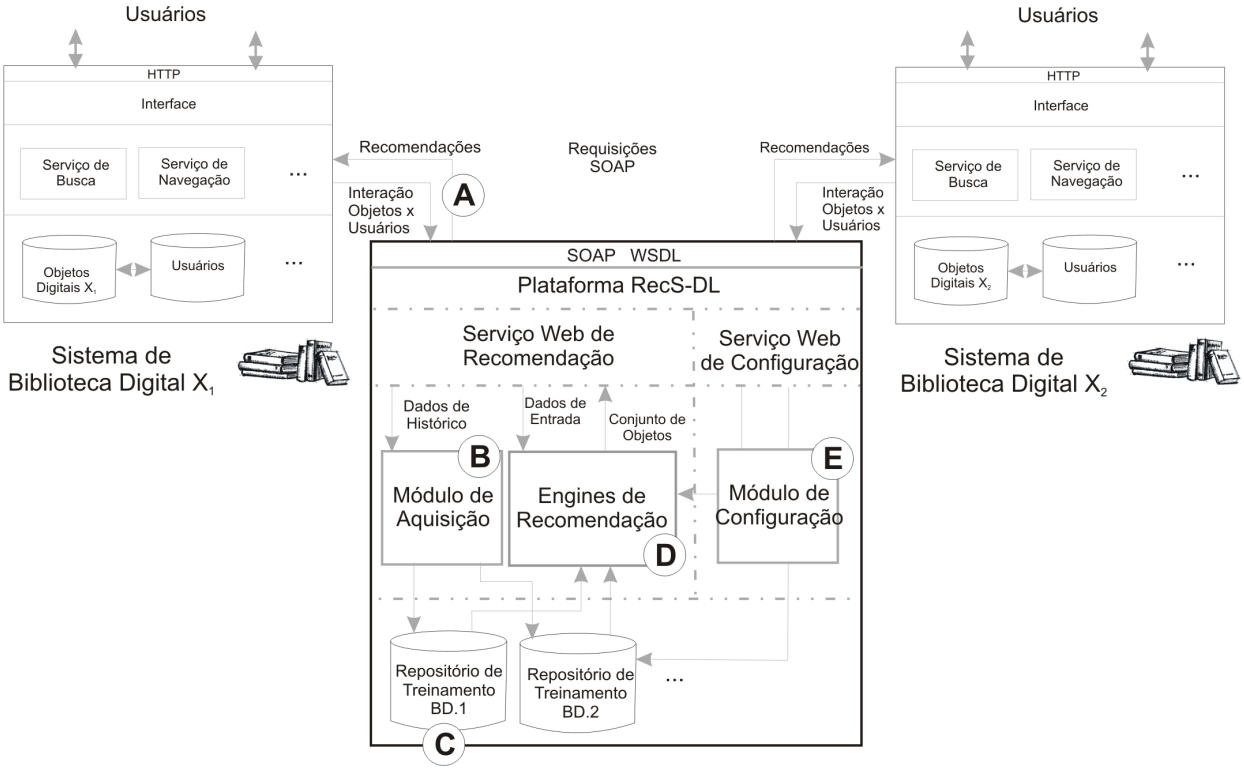


Figura 3.1: Arquitetura Geral da Plataforma.

São representadas na figura duas bibliotecas digitais realizando requisições de recomendação à Plataforma RecS-DL. Como é possível observar, essas aplicações estão isoladas da plataforma, mantendo características e tecnologias próprias e comunicando-se por meio de uma *interface* de requsições de serviços.

Para a *interface* entre a plataforma e as bibliotecas digitais clientes, foi adotada a tecnologia de Serviços Web. Tal tecnologia opera de forma que os métodos de acesso aos serviços da plataforma são definidos segundo o padrão WSDL [13] e a chamada dos serviços por meio de requisições SOAP [27], ambos baseados em XML. Assim, torna-se possível que bibliotecas digitais implementadas em linguagens e sistemas operacionais diversificados sejam clientes comuns dos serviços oferecidos pela plataforma. Toda a comunicação opera sobre o protocolo HTTP, permitindo que aplicações em diversas redes realizem requisições. A parte (A) da Figura 3.1 ilustra essa interação.

Haja vista que as aplicações das bibliotecas digitais clientes estão isoladas, a Plataforma RecS-DL deve prover mecanismos de acesso às informações dessas aplicações. Isso é realizado por meio do Módulo de Aquisição, ilustrado na parte (B) da Figura 3.1. As informações obtidas, por sua vez, são armazenadas em repositórios distintos para cada biblioteca digital cliente, como ilustrado na parte (C) da figura.

Utilizando as informações obtidas pelo Módulo de Aquisição, técnicas de recomendação podem ser aplicadas para oferecimento de recomendações às bibliotecas digitais clientes. Isso é realizado pelos *Engines* de Recomendação, ilustrados na parte (D) da Figura 3.1.

Tendo em vista a possibilidade de utilização da Plataforma RecS-DL por diversas bibliotecas digitais, um grande número de configurações podem ser personalizadas. Essas ações são realizadas no Módulo de Configuração, ilustrado na parte (E) da Figura 3.1.

A Figura 3.2 ilustra como essa arquitetura é implementada na prática. São seguidos basicamente os padrões de uma aplicação Java convencional: a pasta *src* contém os códigos fonte da plataforma; a pasta *lib* as bibliotecas utilizadas e as pastas *build* e *dist* são utilizadas no processo de compilação e empacotamento da plataforma. A pasta *engines* armazena os *engines* de recomendação e respectivos metadados. A pasta *dl-files* contém arquivos de configuração das bibliotecas digitais clientes.

O arquivo *build.xml* contém um *script* utilizado para compilação e empacotamento da plataforma. O arquivo *services.xml* é utilizado para descrição dos serviços oferecidos, contendo todos os métodos e respectivo parâmetros. O arquivos *recomenderws-config.properties* é utlizado para armazenar configurações gerais da Plataforma RecS-DL.

Descritas as principais interações e a estrutura física da plataforma, a Figura 3.3 ilustra um diagrama de casos de uso representando, a nível abstrato, as funcionalidades dos serviços oferecidos pela Plataforma RecS-DL. Dessa forma, esse diagrama apresenta repostas a um dos desafios de pesquisa encontrados: quais funcionalidades deveriam ser oferecidas pela plataforma.

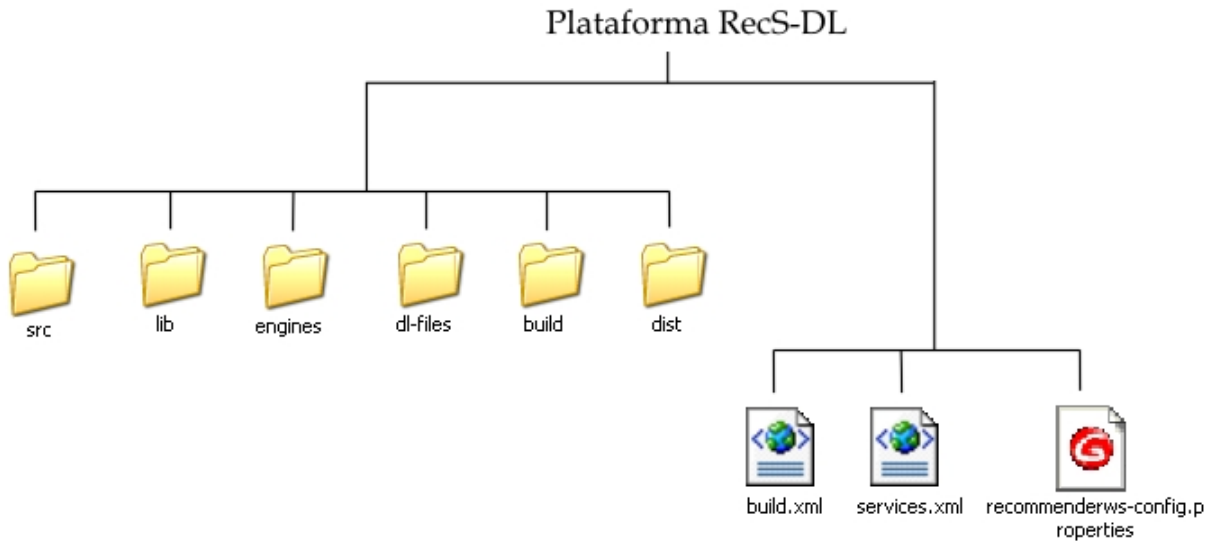


Figura 3.2: Estrutura de arquivos da Plataforma RecS-DL.

As funcionalidades oferecidas são divididas em dois serviços: o Serviço de Recomendação e o Serviço de Configuração. Nas seções seguintes serão discutidos em detalhes o Serviço de Recomendação e o Serviço de Configuração da plataforma, assim como cada método que implementa os casos de uso ilustrados no diagrama da Figura 3.3.

3.4 Serviço de recomendação da Plataforma RecS-DL

Este serviço é reponsável por todos os procedimentos compreendidos no processo de recomendação: a aquisição de dados pela plataforma, a construção de modelos de recomendação e, por fim, a requisição de um conjunto de recomendações. É composto por dois módulos: Módulo de Aquisição e *Engines* de Recomendação. Esses módulos serão discutidos em maiores detalhes nas Seções 3.4.1 e 3.4.2.

3.4.1 Módulo de aquisição

Dado que as técnicas de recomendação baseiam-se principalmente em dados de histórico de comportamento dos usuários, a plataforma dispõe de um módulo de aquisição de dados de treinamento. O objetivo desse módulo é prover mecanismos para acesso aos dados das biblioteca digitais clientes. As interações desse módulo na plataforma são ilustradas na

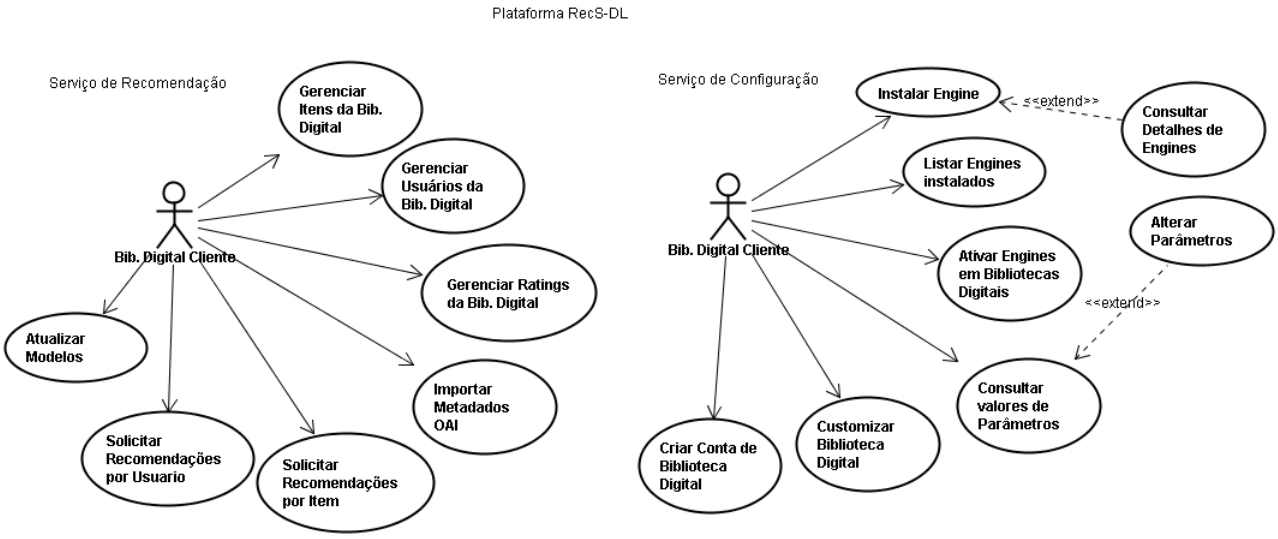


Figura 3.3: Serviços da Plataforma RecS-DL.

parte (B) da Figura 3.1.

As principais classes responsáveis pela implementação do Módulo de Aquisição são exibidas no diagrama de classes da Figura 3.4. São basicamente classes que implementam o *design pattern* DAO (*Data Access Object*), para as entidades da plataforma (*User, Item, Rating*). Essas classes centralizam o acesso a base de dados para gerenciamento da persistência das entidades.

As informações gerenciadas pelo Módulo de Aquisição são armazenadas em um SGBD, para posterior utilização pelos *engines* e respectivas técnicas de recomendação. Este módulo está preparado para operar sobre SGBDs diversos, atendendo também aos requisitos de portabiliade de tecnologia. A implementação foi realizada utilizando apenas tipos de dados e comandos SQL sedimentados em vários SGBDs, possibilitando que a plataforma possa operar com outro gerenciador de banco de dados apenas efetuando a troca do *driver* JDBC. A possibilidade de acesso a repositórios de treinamento diversificados também é ilustrada na parte (C) da Figura 3.1.

O conjunto de dados armazenado pela plataforma em bancos de dados são ilustrados no Diagrama de Classes apresentado na Figura 3.5. Esse modelo de dados é armazendo da mesma maneira (conforme descrito no Anexo A), mas em um repositório distinto para cada biblioteca digital cliente da plataforma. Dessa forma, são gerenciados dados de Usuários, Itens, *Ratings*, *Engines* e seus parâmetros de configuração.

Dado que a padronização do conjunto de metadados é uma questão importante para a interoperabilidade, adotou-se o formato Dublin Core [17] para a tarefa de seleção das

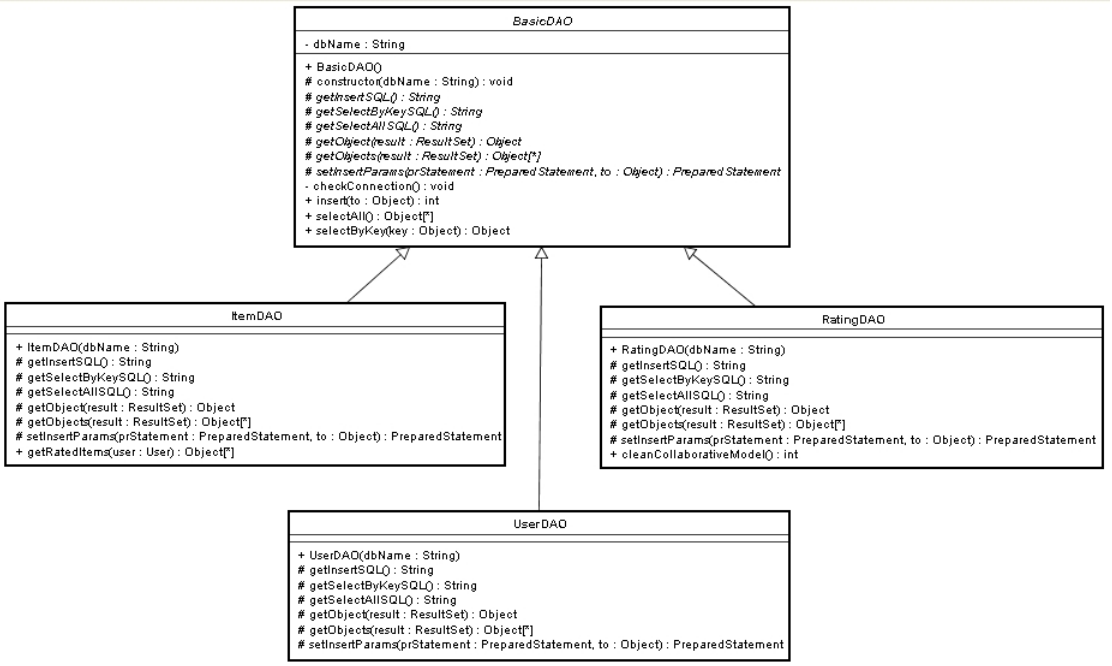


Figura 3.4: Principais classes do Módulo de Aquisição.

informações armazenadas. Este padrão é bastante difundido na literatura e define um conjunto mínimo de elementos capazes de descrever objetos digitais. O Dublin Core define um conjunto de 15 elementos de metadados (Dublin Core Metadata Element Set - DCMES), sendo todos recomendados e nenhum obrigatório [16]. Esses elementos descrevem um objeto através do título, do criador, do assunto, da descrição, do tipo, do formato, entre outros.

Os itens da Plataforma RecS-DL possuem todos os 15 atributos, mais dois atributos para armazenamento de conteúdo, seja textual ou binário: *textcontents* e *base64bincontents*. Dessa forma, os itens podem armazenar tanto documentos textuais quanto imagens, vídeos, mapas, que também podem ser analisados por *engines* visando a realização de recomendações.

O gerenciamento de dados de usuários também dispõe de funcionalidades para armazenamento de informações que podem ser utilizados por *engines* de recomendação. Dados demográficos de interesse e outras informações que compõem o perfil do usuário são armazenados e podem ser utilizados para composição de recomendações personalizadas.

A entidade *rating* armazena as informações que relacionam os usuários e itens de uma biblioteca digital. O relacionamento pode representar desde um histórico de acesso a uma avaliação dada por usuário a um item. Essa informação é representado na plataforma por

um valor numérico inteiro positivo.

De maneira geral, as funcionalidades oferecidas pelo Módulo de Aquisição consistem na manutenção das principais entidades da plataforma. Isso se traduz em métodos para inserção e consulta de itens, usuários e *ratings*.

O Módulo de Aquisição oferece três maneiras diferentes para importação de dados, com o objetivo de atender necessidades diversas de várias bibliotecas digitais. São elas:

- **Serviços Web:** são disponibilizados vários métodos de Serviços Web para inclusão e consulta das principais entidades da plataforma (Usuário, Item e *Rating*). As bibliotecas digitais clientes podem realizar chamadas a esses métodos de Serviços Web para importação dos dados pela Plataforma RecS-DL.

- **Visões:** quando a biblioteca digital cliente for implementada com um banco de dados relacional, a Plataforma RecS-DL pode ser configurada para acessar as informações diretamente no banco de dados da biblioteca digital. Para isso devem ser criadas visões de banco de dados com a estrutura determinada pela Plataforma RecS-DL. Essa estrutura é apresentada no Anexo A.

- **OAI:** é possível realizar a importação de metadados por meio do protocolo OAI (*Open Archives Initiative*) [38]. O protocolo OAI consiste em um conjunto de padrões para requisições HTTP, visando o compartilhamento de metadados entre bibliotecas digitais.

3.4.2 *Engines* de recomendação

Os *engines* consistem em componentes que implementam técnicas de recomendação ou encapsulam ferramentas de recomendação de terceiros, desenvolvidos segundo padrões e *interfaces* definidas. O funcionamento desses elementos assemelha-se ao conceito de *plugin*, já bastante difundido em computação, nos quais módulos de software podem ser desenvolvidos e facilmente instalados sobre uma aplicação principal.

Tal conceito torna a plataforma extensível sem, no entanto, afetar a forma de comunicação e requisição de serviços junto às bibliotecas digitais clientes. Outra vantagem importante consiste na independência em relação à técnica de recomendação utilizada, haja vista que as implementações encontradas na literatura são em geral atreladas a uma técnica específica.

Os *engines* de recomendação têm acesso direto a toda API disponibilizada pela Plataforma RecS-DL, incluindo o Módulo de Aquisição, Módulo de Configuração e métodos da classe *BasicEngine*, que serão apresentados a seguir. Dessa forma os *engines* podem realizar operações como: (i) acessar as informações das bibliotecas digitais clientes por

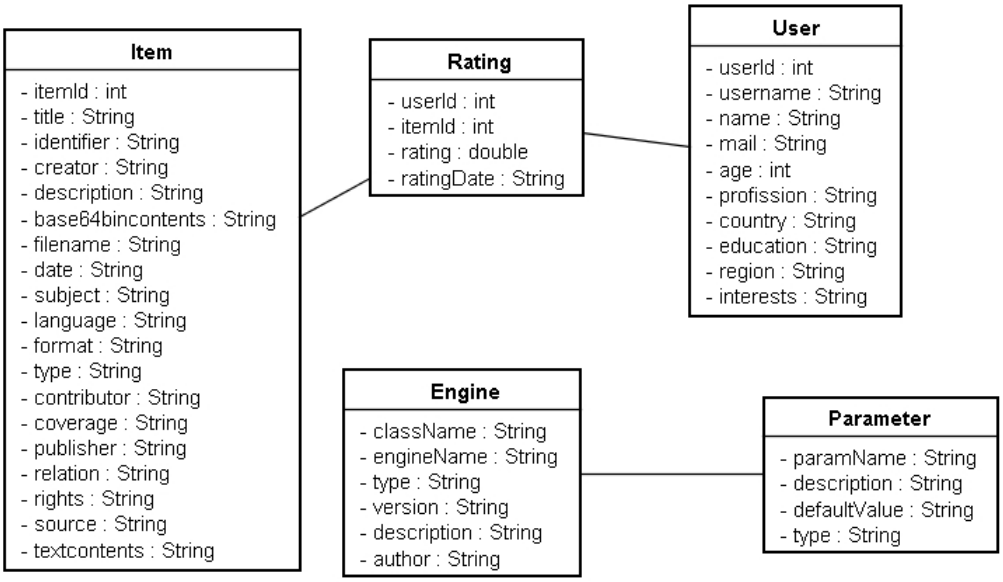


Figura 3.5: Diagrama de Classes da Plataforma armazenados em cada Biblioteca Digital.

meio das classes DAOs do Módulo de Aquisição; (ii) recuperar valores de parâmetros utilizando método da classe *BasicEngine*; (iii) recuperar uma conexão para realizar acesso direto ao repositório de treinamento.

Um cenário de requisição de recomendação e utilização de *engines* é representado no diagrama de sequência apresentado na Figura 3.6: uma biblioteca digital faz uma requisição à plataforma, por meio da classe *RecommenderWS*. Essa classe faz uma requição à classe *EngineFactory*, que por sua vez retorna uma instância de um *engine* de recomendação. Por fim, é realizada a chamada ao método do *engine* que retorna uma lista de itens recomendados. Nesse diagrama, os *engines* são representados pela classe *BasicEngine*, que será discutida adiante.

Os *engines* de recomendação são compostos basicamente por dois elementos: um pacote de implementação e um conjunto de metadados sobre os *engines*. A Figura 3.7 ilustra a estrutura de um *engine*, exibindo parte de um conjunto de metadados e os métodos abstratos que devem ser implementados pela classe de implementação principal. Os detalhes de construção dos *engines* serão discutidos nas subseções seguintes.

Padrões de Interface dos *Engines*

A estrutura para implementação de *engines* definida pela plataforma concentra-se na implementação de uma classe principal do *engine*. Essa classe deve estender a classe

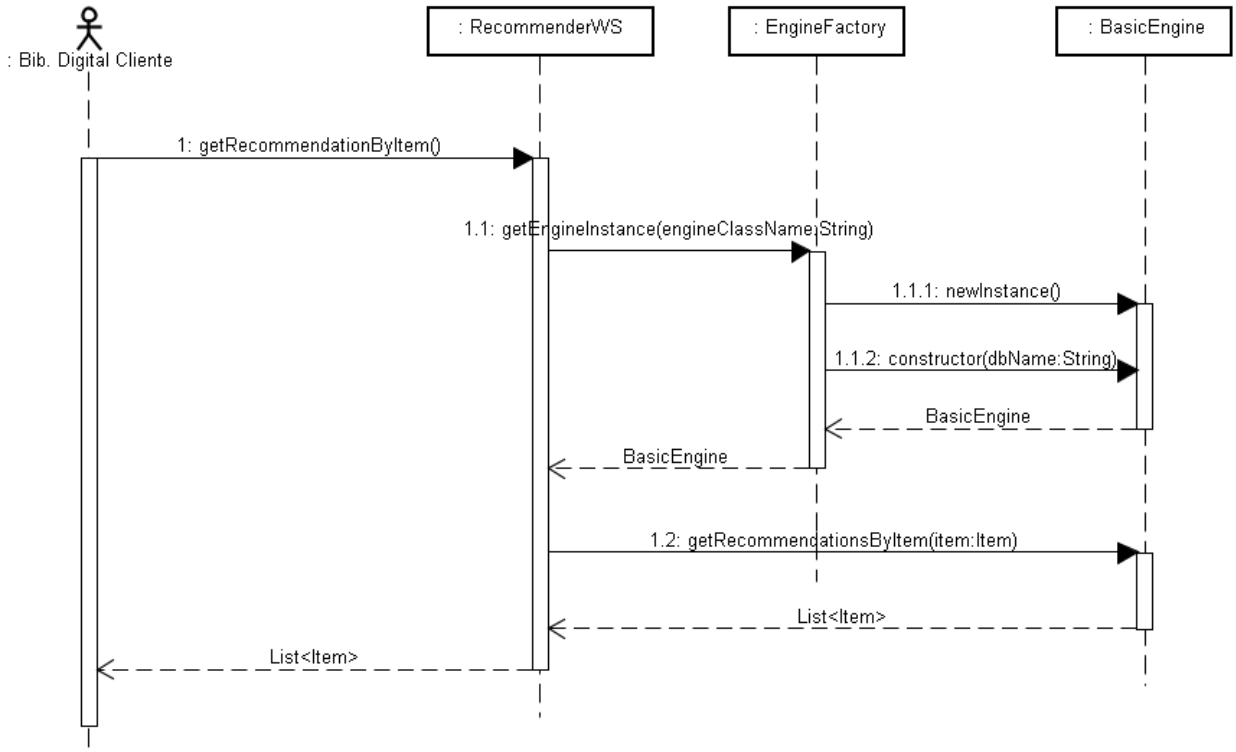


Figura 3.6: Diagrama de Sequência de uma requisição aos *Engines* de Recomendação.

abstrata *BasicEngine*, definida pela plataforma, e implementar os métodos de acesso às técnicas de recomendação, ilustrados na parte (C) da Figura 3.7.

Os seguintes métodos devem ser obrigatoriamente implementados pela classe principal de implementação:

- *makeModel*: método de atualização do *modelo* de treinamento. Aqui o termo *modelo* visa designar quaisquer estruturas que tenham como objetivo armazenar dados produzidos pelos algoritmos de treinamento das técnicas de recomendação. Os engines de recomendação podem armazenar seus modelos de treinamento em banco de dados relacionais ou no sistema de arquivos;

- *getRecommendationsByItem*: retorna um conjunto de recomendações para um determinado item;

- *getRecommendationsByUser*: retorna um conjunto de recomendações para um determinado usuário;

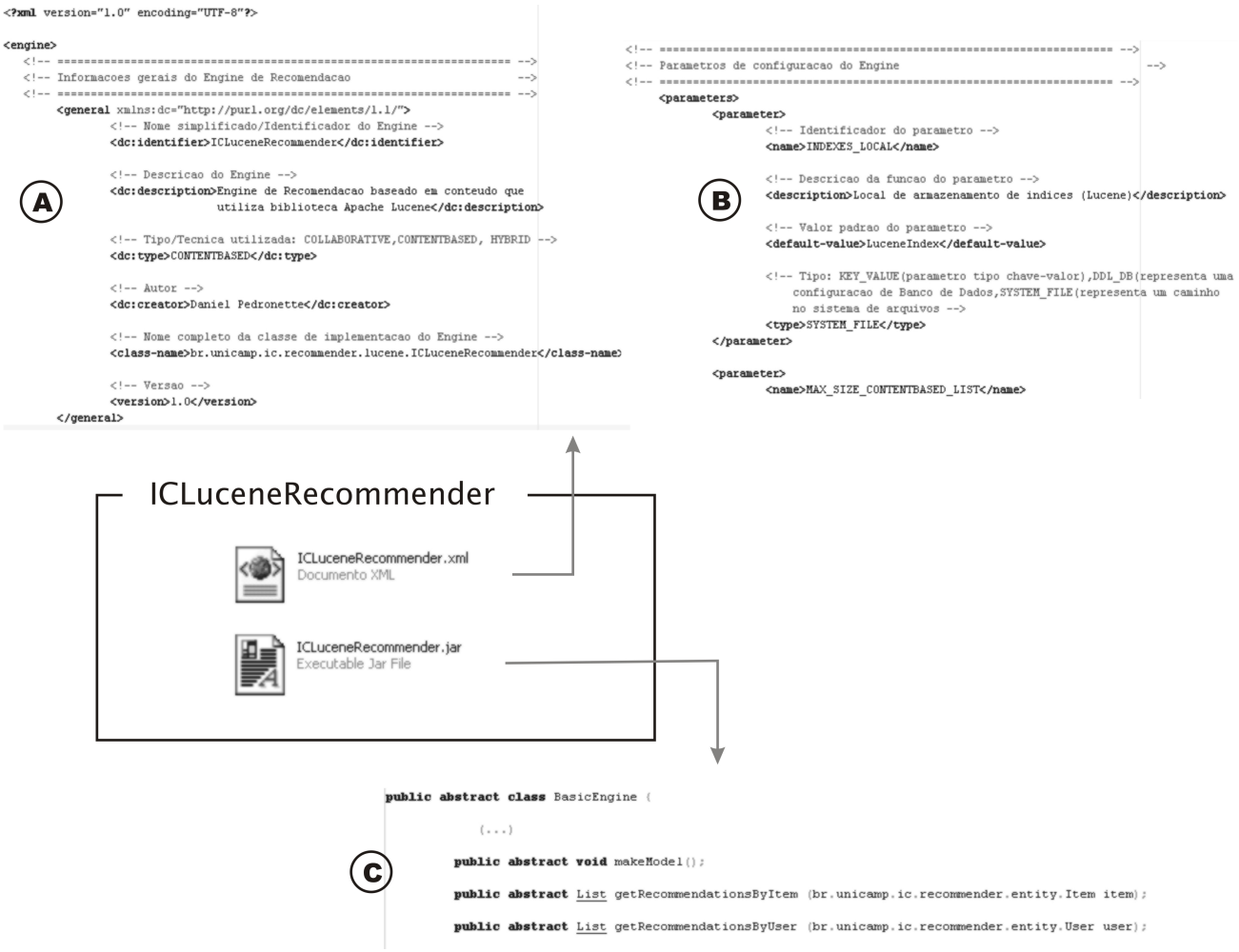


Figura 3.7: Composição dos *Engines*.

A classe *BasicEngine* também implementa métodos de acesso a serviços providos pela plataforma como:

- *getConfigParameter*: dada uma identificação do parâmetro e da biblioteca digital, retorna o valor armazenado do parâmetro. Os parâmetros armazenam configurações que podem variar entre as várias bibliotecas digitais clientes, conferindo flexibilidade à plataforma;

- *getEngineInstance*: dada um nome de um *engine* de recomendação retorna uma instância desse *engine*. Esse método foi modelado visando a construção de *engines* híbridos, que podem instanciar vários outros engines e combinar seus resultados.

- *getConnection*: retorna uma conexão do SGBD, possibilitando o acesso direto ao banco de dados pelo *engine*.

A implementação do *engine* deve ser dada por uma biblioteca *.jar*, da tecnologia Java. Essa biblioteca deve conter a implementação do *engine* propriamente dita, composta de classes e bibliotecas.

Todavia, é importante salientar também que, embora a implementação do *engine* seja dada na tecnologia Java, não há impedimentos para que um *engine* faça chamadas a módulos em outras linguagens e/ou tecnologias, utilizando requisições comuns (como SOAP). Assim, para criar um mecanismo de recomendação baseado em computação distribuída, por exemplo, basta desenvolver um engine que efetue tais requisições entre tecnologias, e a plataforma continuará compatível com as interfaces definidas.

Metadados dos *Engines*

O conjunto de metadados consiste em arquivos XML contendo as principais informações de criação e configuração dos *engines*. As informações contidas nos descritores são dividas em dois grandes grupos, como ilustrado na Figura 3.7: Informações Gerais, parte (A); e Parâmetros de Configuração, parte (B).

O conjunto de informações gerais contém metadados como identificador, descrição, autor, e o nome da classe principal de implementação do *engine*. Utilizando essas informações, a plataforma pode instanciar e executar os *engines*.

Para a composição das informações gerais dos metadados, foram utilizados alguns dos 15 elementos do Dublin Core, identificados no descritor pelo *Namespace* (*dc:*), segundo recomendações do padrão Dublin Core para documentos XML [57]. Outros elementos foram adicionados, para informações necessárias à plataforma, como classe de implementação e versão do *engine*.

O conjunto de metadados deve conter ainda informações sobre o conjunto dos parâmetros de configuração utilizados pelo *engine* de recomendação. Os parâmetros têm como objetivo tornar os *engines* flexíveis diante de bibliotecas digitais distintas.

Para cada parâmetro deve haver uma identificação, uma breve descrição e um valor padrão. Esses parâmetros, por sua vez, serão copiados para a base de dados de cada biblioteca digital cliente, onde poderão ser configurados e customizados.

A Figura 3.8 exibe o DTD que define a estrutura para os metadados XML.

3.5 Serviço de configuração da Plataforma RecS-DL

Dada a grande versatilidade do conceito de *engines*, verificou-se a necessidade da disponibilização de serviços de configuração da plataforma. Dessa forma, foi criado um Serviço

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

<!ELEMENT engine (parameters|general)*>

<!ELEMENT general (dc:creator|dc:identifier|dc:description|dc:type|version|class-name)*>
<!ATTLIST general xmlns:dc CDATA #IMPLIED>
<!ELEMENT dc:identifier (#PCDATA)>
<!ELEMENT dc:description (#PCDATA)>
<!ELEMENT dc:type (#PCDATA)>
<!ELEMENT dc:creator (#PCDATA)>
<!ELEMENT class-name (#PCDATA)>
<!ELEMENT version (#PCDATA)>

<!ELEMENT parameters (parameter)*>
<!ELEMENT parameter (type|default-value|description|name)*>
<!ELEMENT name (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT default-value (#PCDATA)>
<!ELEMENT type (#PCDATA)>
```

Figura 3.8: DTD definido para os descritores XML.

Web complementar, contendo métodos específicos de configuração. Este serviço é de vital importância para a flexibilidade da plataforma, garantindo que ela possa ser adaptada e configurada de acordo com as necessidades de cada biblioteca digital.

A Figura 3.9 ilustra um trecho do documento XML que especifica os métodos dos Serviço de Configuração da plataforma. Na parte (A) são definidos o nome do Serviço Web e a classe que implementa o serviço. As partes (B) e (C) ilustram a definição de um dos métodos desse serviço: na parte(B) é definido o nome do método (*installEngine*) e na parte (C) são definidos os seus parâmetros (*engineName, xmlFile, jarEncodedFile*).

Serão discutidos a seguir o funcionamento dos principais métodos oferecidos pelo Serviço de Configuração.

3.5.1 Instalação de *engines*

O método *installEngine* possibilita que um engine implementado seja instalado na plataforma. O método de instalação de *engines* recebe três parâmetros de entrada: a biblioteca de implementação, os metadados XML e o nome (identificador) do engine, que também consta no arquivo XML. Os arquivos devem ser submetidos ao serviço codificados em formato Base64. O formato Base64 permite representar arquivos binários de forma textual, evitando que os Serviços Web tenham de operar com arquivos anexos. Quando uma requisição de instalação de *engine* é realizada na plataforma, os seguintes procedimentos são executados, conforme é ilustrado na Figura 3.10:

1. o arquivo de metadados XML é copiado para a pasta de descritores da plataforma;


```
<service name="RecommenderWSConfig">
    <description>
        Configuração do Serviço Web de Recomendação
        Desenvolvido no Instituto de Computação (IC) da Universidade Estadual de Campinas (Unicamp)
    </description>
    <parameter name="ServiceClass" locked="false">br.unicamp.ic.recommender.webservice.RecommenderWSConfig</parameter>

    <!-- ===================================================================== -->
    <!--  Recursos de Instalação dos Engines no RecommenderWS                  -->
    <!-- ===================================================================== -->

    <operation name="installEngine">
        <messageReceiver class="org.apache.axis2.rpc.receivers.RPCMessageReceiver" />
        <parameter name="engineName" locked="false">
        </parameter>
        <parameter name="xmlFile" locked="false">
        </parameter>
        <parameter name="jarEncodedFile" locked="false">
        </parameter>
    </operation>
```

Figura 3.9: Métodos oferecidos pelo serviço de configuração da plataforma.

2. a biblioteca Java (.jar) é copiada para a pasta de *engines*;

3. as bibliotecas internas ao arquivo da biblioteca Java (.jar) são copiadas para a pasta lib da plataforma;

4. realizam-se a compilação e o empacotamento da plataforma já com as novas bibliotecas;

5. é realizada a instalação (*deploy*) da plataforma no servidor de aplicação.

Dado que o procedimento de instalação pode durar muito tempo e causar o *timeout* da operação, no momento da criação da requisição HTTP, é criado um processo no servidor que continua executando independente da requisição. Dessa forma, o procedimento de instalação é assíncrono, ou seja, um retorno é enviado antes que o procedimento tenha efetivamente terminado. Assim uma resposta é retornada ao cliente, informando de que o processo iniciou corretamente sua operação.

3.5.2 Criação de contas de bibliotecas digitais:

A Plataforma RecS-DL pode ser acessada/utilizada por várias bibliotecas digitais clientes de forma simultânea. Isso é permitido a partir do uso do conceito de contas para bibliotecas digitais clientes. Dessa forma, uma biblioteca digital que deseja utilizar os serviços da plataforma, primeiro deve registrar-se (*criar uma conta*).

O procedimento de registro na plataforma consiste basicamente na criação de um espaço de armazenamento para biblioteca digital. O procedimento cria um local no sistema

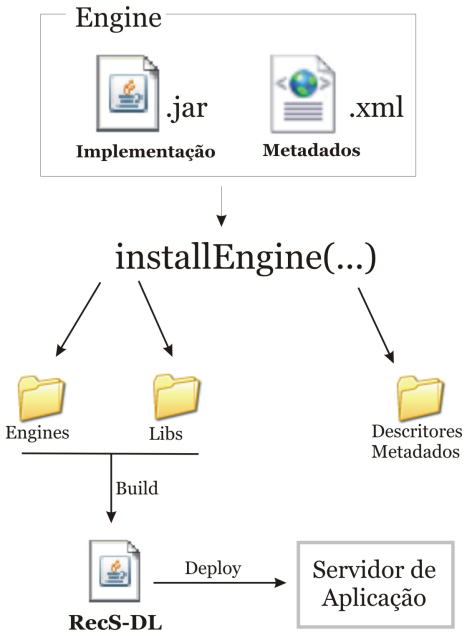


Figura 3.10: Procedimentos de instalação de um *engine* de recomendação

de arquivos utilizado para armazenamento de arquivos de configuração e dados gerados pelos *engines* de recomendação.

Esse espaço contém inicialmente um arquivo de configuração que indica as informações de conexão ao SGBD da biblioteca digital. Sob esse aspecto, há duas maneiras diferentes de criar uma conta para uma biblioteca digital:

1. **Criação de novo repositório:** utiliza as informações de conexão de banco de dados padrão da plataforma para criar um novo banco de dados e toda a estrutura de tabelas definida pela plataforma.

2. **Configuração de acesso a repositório existente:** considera que a estrutura de banco de dados já esteja criada, e apenas configura a conexão de acesso ao banco de dados.

As duas opções foram viabilizadas com o objetivo de tornar a plataforma mais flexível. Para uma biblioteca digital que não deseja expor diretamente o acesso ao seu banco de dados ou estruturas de armazenamento, é interessante que se crie uma nova estrutura de banco de dados e utilizem-se os métodos do módulo de aquisição para submissão das informações da biblioteca digital.

Em contrapartida, se não houver problemas em prover acesso direto ao banco de dados, é possível criar visões de dados compatíveis com as estruturas da Plataforma RecS-DL e apenas configurar as informações de conexão a esse banco de dados.

Os comandos DDL (*Data Definition Language*) - SQL de criação das estruturas de banco de dados da Plataforma RecS-DL são listados no Anexo A dessa dissertação.

3.5.3 Ativação de *engines* e ajuste de parâmetros:

Dado que a plataforma pode operar com várias bibliotecas digitais distintas e com características particulares, é possível que um *engine* de recomendação apresente melhores resultados em uma determinada biblioteca do que em outra. Dessa forma, é relevante que seja possível escolher e configurar os *engines* de recomendação que serão utilizados por cada biblioteca digital.

Em virtude disso, foi criado o conceito de ativação de *engines* e ajuste de parâmetros. Uma biblioteca digital cliente deve ativar um engine de recomendação antes de usá-lo efetivamente. O procedimento de ativação consiste em uma cópia dos parâmetros padrões de um engine para uma biblioteca digital. A partir da ativação, essa biblioteca está apta a utilizar esse engine e configurá-lo de acordo com suas necessidades.

As configurações padrões dos *engines* estão contidas nos decritores XML. Após a ativação, essas configurações podem ser modificadas de maneira diferente para cada biblioteca digital, garantindo flexibilidade à plataforma.

3.6 Protótipo

Um protótipo foi construído visando validar por meio da implementação os requisitos e projeto previamente discutidos nas seções anteriores. O desenvolvimento do protótipo consistiu na implementação de quatro componentes:

- **Plataforma RecS-DL:** implementação da Plataforma RecS-DL propriamente dita, dada pelos métodos oferecidos pelos Serviços de Recomendação e Configuração, como descritos nas Seções anteriores;

- ***Engines* de Recomendação**: implementação de *engines* de recomendação para possibilidade de teste efetivo dos métodos da plataforma e da estrutura de *engines* construída;

- **Aplicação Cliente:** construção de uma aplicação cliente com o objetivo de simular o comportamento de uma bibliteca digital cliente, acessando os métodos dos Serviços Web da plataforma.

- **Ferramenta de instalação para a plataforma:** construção de uma aplicação responsável por instalar a Plataforma RecS-DL, Aplicação Cliente e os softwares servidores necessários a execução dessas aplicações.

A Figura 3.11 ilustra o protótipo construído e seus componentes. A Parte (A) ilustra a Plataforma RecS-DL e seus respectivos serviços. A Parte (B) apresenta os *engines* como itens acopláveis à plataforma. Por fim, a Parte (C) apresenta a aplicação cliente construída. O processo de construção de cada um dos componentes será discutido em mais detalhes a seguir.

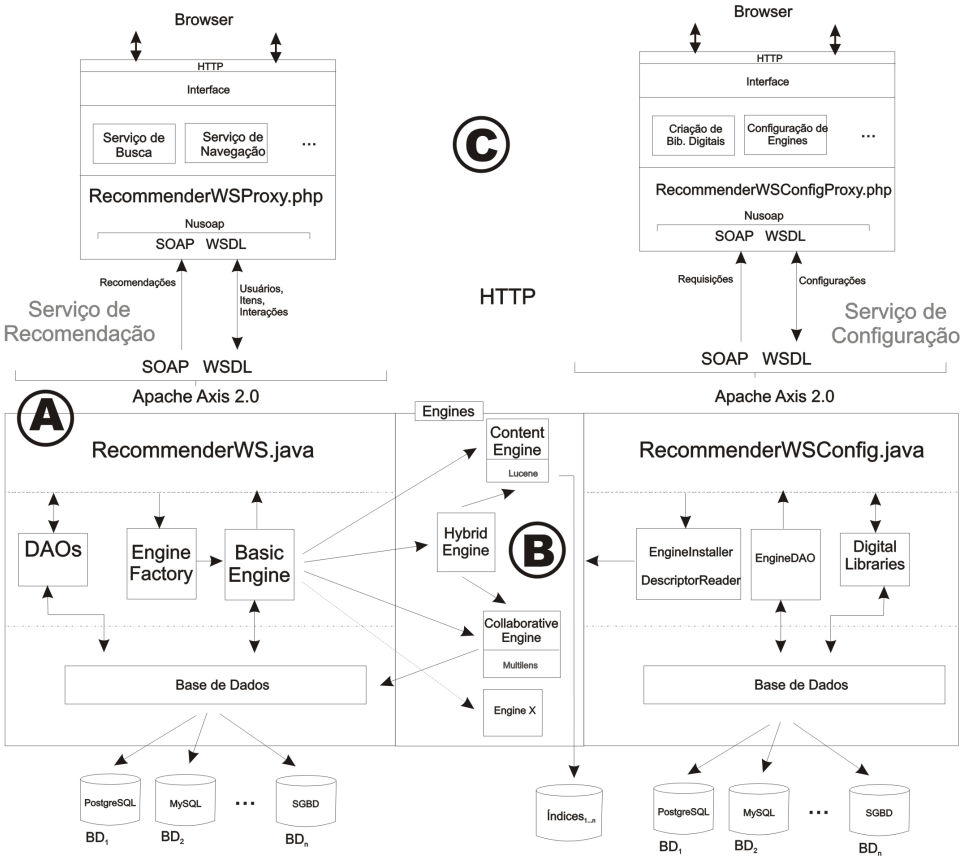


Figura 3.11: Arquitetura do protótipo construído.

Sob o aspecto tecnológico, foram testadas diversas ferramentas baseadas em software livre. O pacote Apache Axis [3] foi utilizado para prover a interface de Serviço Web, descrições WSDL e requisições SOAP.

Os testes foram realizados selecionando-se um conjunto de serviços da plataforma e executando-os sobre os mesmos dados, mas em ferramentas distintas. Os procedimentos

selecionados foram requisições de recomendação para um *Usuário*, para um *Item* e a consulta a um *Item*.

Quanto aos servidores de aplicação, foram selecionados para os testes os servidores JBoss [59] e Apache Tomcat [4], dada a ampla utilização destes no mercado. Em todos os procedimentos a plataforma comportou-se da mesma forma em ambas as ferramentas atendendo aos requisitos de portabilidade tecnológica. Em relação aos SGBDs, foram selecionados o MySQL [6] e PostgreSQL [7], também ferramentas de software livre amplamente utilizadas e nos quais a plataforma apresentou comportamento idêntico.

Haja vista que a plataforma foi desenvolvida utilizando a tecnologia Java, o protótipo procurou validar também a independência de linguagem do aplicativo de biblioteca digital cliente. Assim, o cliente do protótipo foi desenvolvido utilizando linguagem PHP, e a biblioteca de *WebServices* NuSoap [48] sobre servidor Apache.

3.6.1 *Engines* de recomendação

O protótipo procurou também validar a independência de técnicas de recomendação. Tal independência é conseguida na plataforma a partir da arquitetura de *engines* proposta. Dessa forma, foram desenvolvidos e instalados na plataforma três *engines* de recomendação, cada um deles baseado numa técnica de recomendação distinta. São eles:

- **ICMultilensRecommender:** *engine* de recomendação baseado em técnicas colaborativas. Utiliza a biblioteca de recomendação Multilens [43], baseando-se no modelo vetorial para métrica de similaridade entre preferências de usuários. Todas as configurações utilizadas para construção do modelo, como tamanho, quantidade de itens mais semelhantes, entre outros, foram transformados em parâmetros do *engine*, passíveis de configurações específicas de cada biblioteca digital.

- **ICLuceneRecommender:** *engine* de recomendação que implementa técnicas baseadas em conteúdo. Utiliza a biblioteca de recuperação de informações Apache Lucene [28] para comparação de conteúdos textuais. As requisições de recomendação para um dado Item, são geradas a partir da similaridade de conteúdo em relação a outros itens. A similaridade de conteúdo é calculada com base na freqüência de termos, implementando o algoritmo TF-IDF [63]. Já para as recomendações de dado um usuário, é recuperado o conteúdo dos itens melhor avaliados por este usuário, e retornados os itens mais semelhantes a este conteúdo.

- **ICHybridRecommender:** implementa uma técnica híbrida de recomendação baseada em pesos. O *engine* realiza uma requisição de recomendação aos dois outros *engines* desenvolvidos, um colaborativo e outro baseado em conteúdo, e combina as listas de recomendação obtidas. Para cada engine é dado um valor configurável, que

representa a relevância (peso) dos resultados desse *engine*. Assim, para todo elemento das listas de itens é atribuída uma pontuação, calculada com base na posição do item na lista e multiplicada pelo peso do *engine* que a produziu. Por fim, os itens que obtiverem as maiores pontuações são retornados como a lista de recomendação final.

3.6.2 Aplicação cliente

Para a aplicação cliente foram construídas *interfaces* Web para todos os métodos dos serviços oferecidos pela plataforma, visando prover meios para testes efetivos de todos os métodos, por meio de uma aplicação Web. Dessa forma, tornou-se possível a simulação de chamadas de bibliotecas digitais clientes à Plataforma RecS-DL.

Ao iniciar a construção da aplicação cliente, foi utilizado o padrão de projeto *Proxy*, criando na aplicação cliente classes que representassem réplica dos serviços oferecidos pela Plataforma RecS-DL. Essas classes são ilustradas na Parte (C) da Figura 3.11, como *RecommenderWSProxy.php* e *RecommenderWSConfigProxy.php*

A Figura 3.12 ilustra a *interface* do cliente Web do protótipo produzido, provendo um menu de acesso às principais funcionalidades oferecidas pela plataforma a partir da aplicação Web construída.

3.6.3 Ferramenta de instalação

Por se tratar de plataforma para oferecimento de serviços, e não somente uma ferramenta, a Plataforma RecS-DL requer a instalação de vários *softwares* para seu completo funcionamento. Essas necessidades foram ainda acentuadas por outras funcionalidades oferecidas pela plataforma, como a instalação de *engines*.

Assim são necessários para o funcionamento da plataforma os seguintes *softwares*:

1. Java (*Java Development Kit*).
2. Ferramenta de scripts (Apache Ant).
3. Servidor de aplicação.
4. Banco de dados relacional.

Esses são os *softwares* necessários exclusivamente para a plataforma. Todavia, as aplicações clientes da platafomra podem demandar outros servidores, como é o caso da aplicação cliente desenvolvida, que requer servidor Apache e linguagem PHP.

Assim sendo, a instalação completa da plataforma e aplicação cliente, com os respectivos *softwares* necessários demanda uma série de procedimentos de instalação e de edição

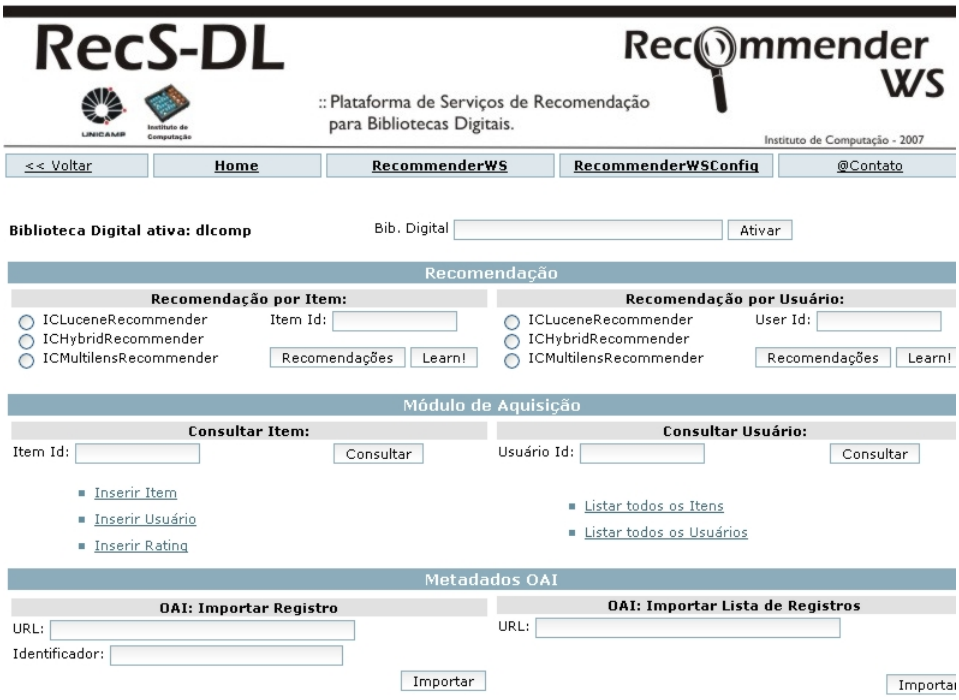


Figura 3.12: Aplicação Web cliente do Protótipo desenvolvido.

de arquivos de configuração. Identificada essa dificuldade, percebeu-se a necessidade da criação de uma ferramenta de instalação para a Plataforma RecS-DL, a aplicação cliente desenvolvida e os respectivos *softwares* necessários. Outra dificuldade identificada foi o controle de execução dos softwares servidores, que na maioria constituem serviços do sistema operacional.

A solução encontrada foi a implementação de uma ferramenta de instalação responsável por instalar e configurar todos os *softwares* necessários, configurar a Plataforma RecS-DL e a aplicação cliente e, por fim, criar uma ferramenta de controle de execução na máquina onde a plataforma foi instalada. O único pré-requisito da ferramenta de instalação consiste na prévia instalação do *Java Development Kit* em um ambiente Windows. Uma versão da ferramenta para ambiente Linux pode ser desenvolvida apenas trocando-se os *softwares* servidores. A Figura 3.13 ilustra a tela inicial desta ferramenta.

A ferramenta encarrega-se de instalar e configurar os seguintes softwares: Apache Ant, Apache Tomcat, PostgreSQL, Servidor Apache e PHP. A Plataforma RecS-DL e a aplicação cliente desenvolvida também são instaladas e configuradas. Por fim, é gerada ferramenta de controle de execução que permite inicializar, parar e testar os *softwares* servidores. A Figura 3.14 ilustra a ferramenta de controle de execução.

Figura 3.13: Ferramenta de instalação da Plataforma RecS-DL.

Figura 3.14: Ferramenta de controle dos *softwares* servidores.

Capítulo 4

Especificação e formalização

Este capítulo apresenta uma especificação formal da plataforma proposta a partir do Arcabouço 5S. Para isso foram propostas novas definições e extensões de conceitos deste arcabouço que são apresentadas a seguir.

4.1 Extensão do conceito de recomendação por meio do Arcabouço 5S

O Arcabouço 5S define todos os principais serviços que compõem uma biblioteca digital. O serviço de recomendação é definido como [23]:

> "**Recommending:** Given a collection and an actor, and a set of ratings for objects in that collection produced by others or the same actor, recommends (produces a subset of that collection) for that particular actor."

Essa definição é bastante adequada quando a recomendação é realizada por meio de técnicas colaborativas. Embora tais técnicas sejam amplamente utilizadas, o termo recomendação tem assumido uma conotação mais ampla, que abrange diversas técnicas. As mais comuns, como discutido no Capítulo 2, são as técnicas colaborativas e baseadas em conteúdo. A seguir, serão discutidas definições que estendem o conceito de recomendação e apresentam uma abordagem mais detalhada dos conceitos relacionados a *recomendação*. São propostas definições diferenciadas de acordo com a técnicas de recomendação: colaborativas e baseadas em conteúdo. Algumas dessas definições posteriormente serão utilizadas para definição da Plataforma RecS-DL.

Recomendação colaborativa: Dada uma coleção, um ator e um conjunto de *ratings* para objetos dessa coleção, produzir um subconjunto da coleção para esse ator particular.

Recomendação baseada no conteúdo: Dada uma coleção e um subconjunto de objetos dessa coleção, produzir outro subconjunto da coleção que se assemelhe ao subconjunto dado.

A Definição 1 apresenta uma formalização do conceito de recomendação, procurando abranger os conceitos de ambos os tipos de técnicas discutidas anteriormente. Na definição apresentada, alguns conceitos do Arcabouço 5S são utilizados, como C e $2^{Colecao}$. Uma coleção C = $\{ObjDig_1, ObjDig_2, . . . , ObjDig_k\}$ é um conjunto de objetos digitais e $2^{Colecao}$ é o número de sub-conjuntos que podem ser formados a partir de uma coleção.

Definição 1. *Um **serviço de recomendação** S_{Rec} é um conjunto de cenários $S_{Rec} = \{srec_1, srec_2, ..., srec_t\}$ onde cada cenário $srec_i$ corresponde a uma seqüência de eventos onde cada evento e_i é associado a uma função OP_{Rec} definida como segue:*

$OP_{rec} : (C \times H_{rec}) \times Sim_s \rightarrow 2^{Colecao}$, *onde:*

- H_{rec}*:* $A \rightarrow H$*, é uma função que mapeia um Ator A a um conjunto de dados de histórico* $H = \{h_1, h_2, ..., h_n\}$*, onde* h_i *identifica um objeto acessado pelo ator A;*

  $Sim_s = OP_{rec}$ *(h,ObjDig)* $\mid h \in H$*,* OP_{rec}*:* $H \times C \rightarrow \mathbb{R}$ *é uma função que associa um número real a um objeto histórico* $h \in H$ *e a um objeto digital ObjDig.*

A função OP_{rec} define o tipo de recomendação provida pelo serviço. Na recomendação colaborativa, o cálculo de OP_{rec} é dado pela aplicação de algoritmos colaborativos ao de histórico de comportamento do usuário. Já na recomendação baseada em conteúdo, o cálculo de OP_{rec} consiste no uso de técnicas recuperação de informações aplicadas aos objetos contidos no histórico do usuário. Em ambos os casos, o número real resultante representa a adequação de dado objeto da coleção aos dados de histórico, onde os que obtiverem os maiores valores compõem o subconjunto recomendado.

4.2 Definição de conceitos de *software* utilizando o Arcabouço 5S

O Arcabouço 5S aborda, sob diversos aspectos, a definição de *Objeto Digital*. Essa definição, por sua vez, está naturalmente relacionada ao conceito de *Stream*, um dos pilares do arcabouço que, de maneira bastante ampla, representa conteúdo.

Todavia, o conceito de conteúdo de *software* não é efetivamente abordado. Serão apresentadas a seguir, algumas definições utilizando os formalismos do Arcabouço 5S, que abordam conceitos de conteúdo para software. Essas definições tem como principal objetivo servir de subsídio para composição de outras definições mais complexas que constituirão a descrição formal da Plataforma RecS-DL.

A Definição 2 apresenta o conceito inicial de *trecho de software executável*, responsável pela implementação do conceito de *evento*, definido pelo Arcabouço 5S. A Figura 4.1 apresenta uma ilustração de um evento de transição e como esse evento relaciona-se com o conceito de *trecho de software executável*, dado pela Definição 2.

Definição 2. *Um* ***trecho de software executável*** $T_s = \{i_1, i_2, ..., i_n\}$ *é uma sequência de instruções de software por meio das quais é implementado um evento* e_s*. (ver Def. 6 [23]).*

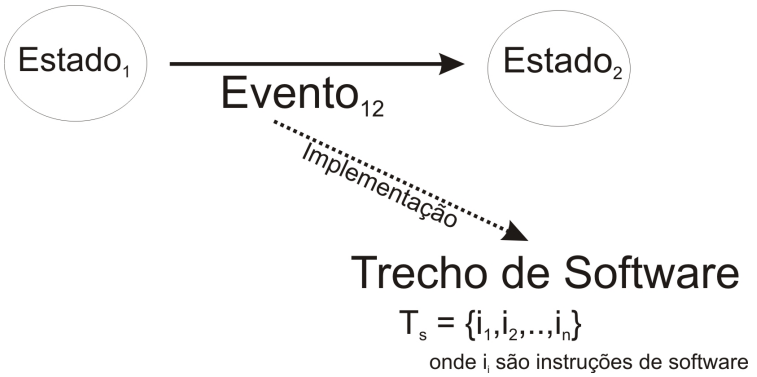


Figura 4.1: Trecho de Software.

A Definição 3 basicamente compõe o conceito de *trecho de software executável* ao conceito de *interface* de dados de entrada e saída de dados para produzir o conceito de *método de software*. Um *método de software* é responsável por implementar um *cenário* de um *serviço*, ambos conceitos definidos pelo Arcabouço 5S.

Definição 3. *Um* ***método de software*** *é uma tupla* $M_s = (H, T, I_e, I_s, I_{me}, F_s, c_s)$ *responsável pela implementação de um cenário* c_s *(ver Def. 7 [23]), onde:*

- H *é um identificador único (rótulo);*
- $T = \{t_1, t_2, ..., t_n\}$ *é um conjunto de trechos de software executáveis;*
- I_e *é uma interface de dados de entrada;*
- I_s *é uma interface de dados de saída;*
- $I_{me} = \{i_{me_1}, i_{me_2}, ..., i_{me_n}\}$ *é um conjunto de interfaces de acesso a métodos externos;*
- F_s*:* $\{T \times I_e \times I_{me}\} \rightarrow I_s$ *é uma função que mapeia os trechos de software executáveis, as interfaces de dados de entrada e métodos externos aos dados de saída.*

A Definição 4 estende o conceito de *objeto digital* definido pelo Arcabouço 5S para produzir o conceito de *objeto digital de sofware*. Para composição desse conceito são utilizados diversos outros conceitos como especificações de metadados, repositórios e métodos de software definido anteriormente.

Definição 4. *Um **objeto digital de sofware** é um objeto digital (para maiores detalhes, ver Def. 16 [23]), $ObjDigSoft$ = (h_{desc},C_{Mt},C_{Md},C_{Rep},C_{Cn}), que atende às seguintes extensões e restrições:*

- $h_{desc} \in H$, *representa um conjunto de identificadores únicos (rótulos);*
- $C_{Mt} = \{mt_1, mt_2, ..., mt_n\}$ *é um conjunto de métodos de software;*
- $C_{Md} = \{md_1, md_2, ..., md_n\}$ *é um conjunto de especificações de metadados descritivos (ver Def. 12 [23]), responsáveis por armazenar informações sobre o software, interfaces e parâmetros de configuração;*
- $C_{Rep} = \{R_1, R_2, ..., R_n\}$ *é um conjunto de repositórios (ver Def. 19 [23]) acessados pelos métodos de software do conjunto C_{Mt}, para armazenamento ou recuperação de informações;*
- $C_{Cn} = \{cn_1, cn_2, ..., cn_n\}$ *é um conjunto de cenários implementado pelos métodos de software do elemento C_{Mt}, e $C_{Cn} \subset Serv$, onde $Serv$ é um serviço definido total ou parcialmente pelo conjunto C_{Cn}.*

Por fim, a Definição 5 define uma *plataforma de software*, composta de um conjunto de serviços e dos elementos utilizados na implementação desses serviços, como objetos digitais de software, repositórios e metadados.

Definição 5. *Uma **plataforma de software** é um ambiente para oferecimento de serviços a partir da execução de objetos de digitais de software, definida por uma tupla P = (C_{Od}, C_{Serv}, C_{Rep}, C_{Md}), onde:*

- C_{Od} *conjunto de objetos digitais de software;*
- C_{Serv} *conjunto de serviços providos pela plataforma e implementados pelo conjunto C_{Od};*
- C_{Rep} *é um conjunto repositórios R (ver Def. 19 [23]), manipulados pela plataforma de software;*
- C_{Md} *é um conjunto de especificações de metadados descritivos sobre os serviços, objetos digitais de sofware e repositórios manipulados.*

4.3 Definição da Plataforma RecS-DL

A especificação da Plataforma RecS-DL exigiu a definição dos serviços, *interfaces* e métodos de acesso. Optou-se por definir formalmente a Plataforma RecS-DL utilizando o Arcabouço 5S, dada a aplicação desse arcabouço no cenário de bibliotecas digitais. Para que essa definição fosse possível, foram utilizados conceitos do Arcabouço 5S e as definições propostas na seção anterior. Serão definidos a seguir os principais serviços e componentes da Plataforma RecS-DL.

Definição 6. Dado que uma plataforma de software é definida como P = (C_{Od}, C_{Serv}, C_{Rep}, C_{Md}), então existe uma Plataforma **RecS-DL**, onde:

- C_{Od} é um conjunto de objetos digitais dado por C_{Od} = (*Módulo de Aquisição*, *Módulo de Configuração*,E_{Rec_1},E_{Rec_2},..., E_{Rec_n}), onde cada E_{Rec_i} representa um *Engine de Recomendação*;
- C_{Serv} é formado pelo conjunto serviços S = *(Serviço de Recomendação, Serviço de Configuração)* providos pela plataforma.
- C_{Rep} representa um conjunto de repositórios configuráveis por meio do Serviço de Configuração. Esses repositórios armazenam dados das bibliotecas digitais que utilizam a plataforma, como configurações e informações de histórico de comportamento dos usuários.
- C_{Md} é dado pelo conjunto M = *(Arquivo de Configurações Gerais da Plataforma, Arquivos de Configuração de Conexões com Repositórios).*

Definição 7. Dado que um *serviço* é um conjunto de *cenários*, então o **Serviço de Recomendação da Plataforma RecS-DL** é um serviço de aquisição e de recomendação que consiste no conjunto de cenários implementados pelo *Módulo de Aquisição* e pelos *Engines de Recomendação* da Plataforma RecS-DL.

Definição 8. Dado que um objeto digital de software é definido como $ObjDigSoft$ = (h_{desc}, C_{Mt}, C_{Md}, C_{Rep}, C_{Cn}), então existe na Plataforma **RecS-DL** o objeto digital de software **Módulo de Aquisição** (ilustrado na parte B da Figura 3.1), onde:

- h_{desc} é "Módulo de Aquisição";
- C_{Md} é dado por um arquivo XML contendo o nome e parâmetros dos métodos. A Figura 4.2 ilustra um trecho desse arquivo;

- C_{Mt} representa a implementação do conjunto C_{Cn}, e é dado pelo conjunto de métodos C_{Mt}= (insertItem, insertRating, getUserById, getAllUsers, getItemById, getAllItems, getRatedItems);

- C_{Cn} é dado pelo conjunto de cenários descritos a seguir (que representa parte do Serviço de Recomendação):

  - **insertItem (item,dbName)**.

    Cenário: $\langle$ e_1 :p = insertItem, e_2 : execute(insertItem), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na inserção do elemento $item$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

  - **insertSerialItem (item,dbName)**.

    Cenário: $\langle$ e_1 :p = insertItem, e_2 : execute(insertItem), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na inserção do elemento $item$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

  - **insertUser (user,dbName)**.

    Cenário: $\langle$ e_1 :p = insertUser, e_2 : execute(insertUser), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na inserção do elemento $item$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

  - **insertRating (rating,dbName).**

    Cenário: $\langle$ e_1 :p = insertRating, e_2 : execute(insertRating), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na inserção do elemento $rating$ no repositório $dbName$; e e_3 representa o evento

de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

- **getUserById (userId,dbName).**
  Cenário: $\langle$ e_1 :p = getUserById, e_2 : execute(getUserById), e_3 : p = response (user)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar um elemento $user$ contido no repositório $dbName$, usando como parâmetro identificador o elemento $userId$; e e_3 representa o evento de retorno da execução onde $p = response(user)$ contém as informações da entidade $user$ cujo identificador é $userId$.
- **getAllUsers (dbName).**
  Cenário: $\langle$ e_1 :p = getAllUsers, e_2 : execute(getAllUsers), e_3 : p = response ($List < user >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo todos os elementos $user$ contidos no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(List < user >)$ contém a lista de entidades $user$.
- **getItemById (itemId,dbName).**
  Cenário: $\langle$ e_1 :p = getItemById, e_2 : execute(getItemById), e_3 : p = response (item)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar um elemento $item$ contido no repositório $dbName$, usando como identificador o parâmetro $itemId$; e e_3 representa o evento de retorno da execução onde $p = response(item)$ contém as informações da entidade $item$ cujo identificador é $itemId$.
- **getAllItems (dbName).**
  Cenário: $\langle$ e_1 :p = getAllItems, e_2 : execute(getAllItems), e_3 : p = response ($List < item >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo todos os elementos $item$ contidos no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de entidades $item$.
- **getRatedItems (userId,dbName).**

Cenário: $\langle$ e_1 :p = getRatedItems, e_2 : execute(getRatedItems), e_3 : p = response ($List < item >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo os elementos $item$ contidos no repositório $dbName$, que estejam relacionados a um elemento $user$ por meio do elemento $rating$, onde o elemento $user$ é identificado pelo parâmetro $userId$; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de entidades $item$ que foram avaliadas pelo usuário $user$.

- **importRecordList (dbName,url)**

  Cenário: $\langle$ e_1 :p = importRecordListl, e_2 : execute(importRecordList), e_3 : p = response (true,false))$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável importar por meio do protocolo OAI [38] os metadados de uma biblioteca digital identificado pelo elemento url para o repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$) define se a execução iniciou corretamente.

- **importRecord (dbName,url,identifier)**

  Cenário: $\langle$ e_1 :p = importRecord, e_2 : execute(importRecord), e_3 : p = response (true,false))$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável importar por meio do protocolo OAI [38] os metadados de um registro identificado pelo elemento $identifier$. Esses metadados são provenientes de uma biblioteca digital identificada pelo elemento url e serão importados para o repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$) define se a execução iniciou corretamente.

- C_{Rep} é dado por $C_{Rep} = (R_{Rel})$, onde R_{Rel} representa um SGBD relacional (ilustrados na parte C da Figura 3.1)) e cuja conexão é configurada pelo Serviço de Configuração da plataforma **RecS-DL**. Esse repositório é responsável por armazenar as principais entidades mantidas pela plataforma, discutidas em maiores detalhes na Seção 3.4.1 e ilustradas no diagrama da Figura 3.5.

Definição 9. Dado que um objeto digital de software é definido como ObjDigSoft = (h_{desc}, C_{Mt}, C_{Md}, C_{Rep}, C_{Cn}, R,$Serv$), então existe na Plataforma **RecS-DL** um conjunto

```
<serviceGroup>
  <service name="RecommenderWS">
    <description>Serviço Web de Recomendação Desenvolvido no Instituto de Computação
      (IC) da Universidade Estadual de Campinas (Unicamp)</description>
    <parameter name="ServiceClass"
      locked="false">br.unicamp.ic.recommender.webservice.RecommenderWS</parameter>
    <!--  ================================================================  -->
    <!--  Métodos de Acesso a Dados                                         -->
    <!--  ================================================================  -->
    <operation name="insertUser">
      <messageReceiver class="org.apache.axis2.rpc.receivers.RPCMessageReceiver" />
      <parameter name="user" locked="false" />
      <parameter name="dbName" locked="false" />
    </operation>
    <operation name="insertItem">
      <messageReceiver class="org.apache.axis2.rpc.receivers.RPCMessageReceiver" />
      <parameter name="item" locked="false" />
      <parameter name="dbName" locked="false" />
    </operation>
                ...
```

Figura 4.2: Metadados dos métodos do objeto digital Módulo de Aquisição.

de objetos digitais de software $E = \{e_{rec1}, e_{rec2}, ..., e_{recn}\}$ (ilustrados na parte D da Figura 3.1), onde cada e_{rec} representa um **Engine de Recomendação**, onde:

- h_{desc} é nome do *Engine de Recomendação*;
- C_{Md} é um conjunto de informações composto por metadados em XML, seguindo o padrão *Dublin Core*. São armazenadas informações sobre o software, versão, autor e parâmetros de configuração;
- C_{Mt} e C_{Cn} são responsáveis pela implementação de um *Serviço de Recomendação* dado pelo seguinte conjunto de métodos e respectivos cenários:
  - **makeModel (engineClassName,dbName)**

    Cenário: $\langle$ e_1 :p = makeModel, e_2 : execute(makeModel), e_3 : p = response (true,false))$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software implementado pela classe $engineClassName$ responsável por atualizar os modelos de treinamento (índices, matrizes, etc) contidos no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$) define se a execução iniciou corretamente.
  - **getRecommendationByUser (user,dbName,engineClassName)**

Cenário: $\langle e_1$:p = getRecommendationByUser, e_2 : execute(getRecommendationByUser), e_3 : p = response $(List < item >)\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software implementado pela classe $engineClassName$ responsável por executar uma técnica de recomendação e gerar uma lista de elementos *item* contidos no repositório $dbName$ que representam um conjunto de recomendações para o parâmetro *user*; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de entidades *item* recomendadas para o usuário *user*.

– **getDefaultRecommendationByUser (user,dbName)**

Cenário: $\langle e_1$:p = getRecommendationByUser, e_2 : execute(getRecommendationByUser), e_3 : p = response $(List < item >)\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software implementado pelo *engine* padrão do repositório $dbName$. Esse método de software implementa uma técnica de recomendação e gera uma lista de elementos *item* contidos no repositório $dbName$, que representam um conjunto de recomendações para o parâmetro *user*; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de entidades *item* recomendadas para o usuário *user*.

– **getRecommendationByItem (item,dbName,engineClassName)**

Cenário: $\langle e_1$:p = getRecommendationByItem, e_2 : execute(getRecommendationByItem), e_3 : p = response $(List < item >)\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software implementado pela classe $engineClassName$, responsável por implementar uma técnica de recomendação e gerar uma lista de elementos *item* contidos no repositório $dbName$ representando um conjunto de recomendações para o parâmetro *item*; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de recomendações para o parâmetro *item*.

– **getDefaultRecommendationByItem (item,dbName)**

Cenário: $\langle e_1$:p = getDefaultRecommendationByItem, e_2 : execute(getRecommendationByItem), e_3 : p = response $(List < item >)\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de

software implementado pelo *engine* padrão do repositório $dbName$. Esse método de software implementa uma técnica de recomendação e gera uma lista de elementos *item* contidos no repositório $dbName$, que representam um conjunto de recomendações para o parâmetro *item*; e e_3 representa o evento de retorno da execução onde $p = response(List < item >)$ contém a lista de recomendações para o parâmetro *item*.

- R é um repositório de treinamento, definido por cada *Engine de Recomendação*. Os repositórios variam de acordo com o engine, desde Sistema de Arquivos a SGBD relacionais.

- $Serv$ é parte do Serviço de Recomendação da plataforma **RecS-DL**.

Definição 10. Dado que um *serviço* é um conjunto de *cenários*, então o **Serviço de Configuração da Plataforma RecS-DL** consiste no conjunto de cenários implementados pelo *Módulo Configuração*, responsável por prover funcionalidades de configuração à Plataforma RecS-DL e respectivas bibliotecas digitais clientes.

Definição 11. Dado que um objeto digital de software é definido como ObjDigSoft = (h_{desc}, C_{Mt}, C_{Md}, C_{Rep}, C_{Cn}, R, $Serv$), então existe na Plataforma **RecS-DL** um objeto digital de software **Módulo de Configuração**, onde:

- h_{desc} é "Módulo de Configuração";

- C_{Md} é dado por um arquivo XML contendo o nome, descrição e parâmetros dos métodos;

- C_{Mt} e C_{Cn} são dados pelos seguinte conjunto de métodos e respectivos cenários:

  - **installEngine (engineName,xmlFile,jarEncodedFile)**

    Cenário: $\langle$ e_1 :p = installEngine e, e_2 : execute(installEngine), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na instalação do objeto digital de $engineName$, cuja implementação é dada pelo elemento $jarEncodedFile$ e o conjunto de metadados pelo elemento $xmlFile$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

– **getParametersList (engineName)**

Cenário: $\langle$ e_1 :p = getParametersList, e_2 : execute(getParametersList), e_3 : p = response ($List < Parameter >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo os paramentros do elemento $engineName$ contidos no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(List < Parameter >)$ contém a lista de entidades *parameter* relacionadas ao engine $engineName$.

– **installedEngines**

Cenário: $\langle$ e_1 :p = installedEngines, e_2 : execute(installedEngines), e_3 : p = response ($List < Engine >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo *engines de recomendação* instalados na *Plataforma RecS-DL*; e e_3 representa o evento de retorno da execução onde $p = response(List < Engine >)$ contém a lista de entidades *engine* que representam os engines instalados.

– **getEngineInformation (engineName)**

Cenário: $\langle$ e_1 :p = getEngineInformation, e_2 : execute(getEngineInformation), e_3 : p = response (engine)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar as informações do elemento *engine* instalado na Plataforma RecS-DL, usando como parâmetro identificador o elemento $engineName$; e e_3 representa o evento de retorno da execução onde $p = response(engine)$ contém as informações da entidade *engine* cujo identificador é $engineName$.

– **activateEngine (engineName,dbName)**

Cenário: $\langle$ e_1 :p = activateEngine, e_2 : execute(activateEngine), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que consiste na ativação do elemento $engineName$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

– **getDefaultEngine (dbName)**

Cenário: $\langle$ e_1 :p = getDefaultEngine, e_2 : execute(getDefaultEngine), e_3 : p = response (engine)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar as informações do elemento *engine* padrão para o respoitório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(engine)$ contém as informações da entidade *engine*.

– **getActivatedEngines (dbName)**

Cenário: $\langle$ e_1 :p = getActivatedEngines, e_2 : execute(getActivatedEngines), e_3 : p = response ($List < Engine >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista contendo *engines de recomendação* ativos no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(List < Engine >)$ contém a lista de entidades *engine* que representam os engines ativos.

– **setDefaultEngine (dbName,engineClassName)**

Cenário: $\langle$ e_1 :p = setDefaultEngine, e_2 : execute(setDefaultEngine), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por tornar padrão o engine de recomendação identificado por $engineClassName$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

– **getParameter (dbName,engineClassName,parameter)**

Cenário: $\langle$ e_1 :p = getEngineInformation, e_2 : execute(getEngineInformation), e_3 : p = response (engine)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar as informações do elemento *engine* instalado na Plataforma RecS-DL, usando como parâmetro identificador o elemento $engineName$; e e_3 representa o evento de retorno da execução onde $p = response(engine)$ contém as informações da entidade *engine* cujo identificador é $engineName$.

– **setParameter (dbName,engineClassName,parameter,value)**

Cenário: $\langle$ e_1 :p = setParameter, e_2 : execute(setParameter), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando

uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software que define como *value* o valor do parâmetro *parameter* relacionado ao engine de recomendação $engineClassName$ no repositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

- **getParameterValues (dbName,engineClassName)**

  Cenário: $\langle$ e_1 :p = getParameterValues, e_2 : execute(getParameterValues), e_3 : p = response ($List < Parameter >$)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por recuperar uma lista de elementos *parameter* relacionados ao engine de recomendação $engineClassName$ no respositório $dbName$; e e_3 representa o evento de retorno da execução onde $p = response(List < Parameter >)$ contém a lista de entidades *parameter* que representam parâmetros de configuração.

- **createDigitalLibrary (dbName)**

  Cenário: $\langle$ e_1 :p = createDigitalLibrary, e_2 : execute(createDigitalLibrary), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável criar uma conta de biblioteca digital e respectivo repositório $dbName$ na Plataforma RecS-DL; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

- **configureDigitalLibrary (dbName,urlDB,driver,user,password)**

  Cenário: $\langle$ e_1 :p = configureDigitalLibrary, e_2 : execute(configureDigitalLibrary), e_3 : p = response (true,false)$\rangle$, onde e_1 é um evento gerado por uma biblioteca digital invocando uma ação em *RecS-DL*, p especifica a ação correspondente que está sendo invocada; e_2 representa a execução do método de software responsável por configurar uma conta de biblioteca digital para o repositório $dbName$ especificando a tupla de parâmetros de configuração dada por $(urlDB, driver, user, password)$; e e_3 representa o evento de retorno da execução onde $p = response(true, false)$ define se a execução foi concluída corretamente.

- R é o repositório de configuração para cada Biblioteca Digital usuária da plataforma **RecS-DL**.

- *Serv* é o Serviço de Configuração da plataforma **RecS-DL**.

Capítulo 5

Validação

Este capítulo tem como objetivo descrever os experimentos realizados e respectivos resultados obtidos. Os experimentos foram definidos visando avaliar a possibilidade de integração e utilização da Plataforma RecS-DL por bibliotecas digitais. Também foram avaliados aspectos de usabilidade e documentação da Plataforma RecS-DL. Foram realizados três experimentos, dois deles envolvendo potencias usuários e um utilizando uma aplicação real, a Biblioteca Digital da Unicamp. A metodologia utilizada é discutida a seguir:

- **Estudos de caso:**

  Os estudos de caso realizados consistiram na verificação funcional dos serviços oferecidos pela Plataforma RecS-DL. Os testes foram realizados utilizando conjuntos de dados de domínios diversos visando verificar também a independência de domínio de aplicação da plataforma.

- **Experimentos com a Biblioteca Digital da Unicamp:**

  Os experimentos realizados com a Biblioteca Digital da Unicamp consistem na adição de funcionalidades de recomendação utilizando a Plataforma RecS-DL. Os procedimentos executados compreenderam, além da instalação e configuração da Plataforma RecS-DL, a integração ao software Nou-Rau, gerenciador da Biblioteca Digital da Unicamp. Detalhes dos procedimentos executados e resultados obtidos são discutidos na Seção 5.2.

- **Experimentos com potenciais usuários:**

  Os experimentos consistem em avaliações realizadas por potenciais usuários da Plataforma RecS-DL. Os pontenciais usuários identificados consistem em estudantes de graduação e pós-graduação do Instituto de Computação da Unicamp, cursando a

disciplina de Bibliotecas Digitais do segundo semestre de 2007. Participaram dos experimentos 11 estudantes, divididos em 5 duplas e um indivíduo. Sabe-se também que um dos estudantes do curso é administrador de bibliotecas digitais reais. Foi preenchido também um questionário visando identificar o perfil dos participantes, conhecimentos sobre tecnologias utilizadas, formação acadêmica, etc. Esse questionário é apresentado no Anexo B.

Os experimentos e avaliações realizadas buscaram avaliar a possibilidade de integração e utilização da Plataforma RecS-DL por bibliotecas digitais. Dessa forma, embora os *engines* de recomendação incluam a implementação de técnicas de recomendação, o objetivo desses experimentos com ponteciais usuários não é avaliar as técnicas ou a qualidade das recomendações, mas sim verificar se a Plataforma RecS-DL atinge seu principal objetivo de aperfeiçoar a interoperabilidade dos sistemas de recomendação.

Os experimentos realizados consistiram na execução de tarefas práticas utilizando a Plataforma RecS-DL e no preenchimento de relatórios e questionários a respeito das tarefas. A metodologia utilizada para efetivação dos experimentos foi composta por 5 etapas, descritas a seguir:

1. **Criação de Biblioteca Digital:** os participantes dos experimentos, divididos em grupos, executaram a tarefa de criar uma biblioteca digital para os Relatórios Técnicos do Instituto de Computação da Unicamp.

   Foram utilizados nessa tarefa *softwares* de grandes projetos mundias da área: Greenstone, Fedora e D-Space. O Greenstone [74] é um *software* de biblioteca digital dos mais populares, desenvolvido pela University of Waikato e financiado pela UNESCO. O Fedora [50] é mantido em esforço conjunto pela Cornell University Information Science e pela University of Virginia Library. O D-Space [69] é fruto de uma colaboração entre MIT Libraries e Hewllet-Packard. Todos são *softwares* baseados na tecnologia Java e licença *open-source*.

   Essa tarefa teve como principais objetivos: (i) certificar-se de que os participantes estavam familiarizados com os conceitos e *softwares* de bibliotecas digitais; (ii) criar uma biblioteca digital que pudesse ser utilizada nas tarefas seguintes.

2. **Palestra sobre serviços de recomendação e a Plataforma RecS-DL:** foram realizadas duas palestras de 35 a 40 minutos para os potencias usuários da plataforma. Durante as palestras, foram apresentados conceitos de serviços de recomendação e as principais características da Plataforma RecS-DL. Por fim, foi realizada uma breve demonstração da plataforma.

3. **Disponibilização e leitura da documentação da plataforma:** um site

contendo material de documentação Plataforma RecS-DL foi disponibilizado para consulta. Foram disponibilizados:

- **Manual:** documento contendo descrição sobre todas as funcionalidades da Plataforma RecS-DL. Este documento apresenta forma de operação e descrição das funcionalidades da plataforma, informações sobre o Serviço Web, sobre a interface Web disponibilizada e sobre o processo de instalação (ferramentas, pré-requisitos, etc).
- **Guia Rápido:** descreve instruções passo a passo para instalação e utilização da plataforma.
- **Relatório Técnico:** relatório contendo introdução teórica, especificação, modelagem e testes iniciais da plataforma.
- **Modelagem UML:** documento contendo diagramas UML considerados mais relevantes para a documentação da ferramenta. Estão contidos nesse documento diagramas de caso de uso, diagramas de classe e diagramas de sequência.
- **Slides:** apresentação de slides utilizada durante a palestra sobre a Plataforma RecS-DL.
- **Publicações:** apresenta os artigos publicados sobre a Plataforma RecS-DL [51, 52].
- **Software:** foram disponibilizados para *download* o código fonte da plataforma, os *engines* de recomendação desenvolvidos, a ferramenta de instalação e a aplicação cliente desenvolvida.

4. **Execução das tarefas estipuladas:** foram estipulados dois conjuntos de tarefas a serem executadas, visando testar e avaliar a utilização da Plataforma RecS-DL pelos participantes. As tarefas foram descritas em formulários apresentados nos Anexos C e D. Todas as tarefas foram executadas em grupos de duas pessoas.

5. **Preenchimento dos questionários:** para cada conjunto de tarefas executados, foram elaborados relatórios e preenchidos questionários pelos participantes. Os relatórios descreveram detalhes das atividades executadas. Os questionários buscaram avaliar a plataforma sob diversos aspectos a partir das tarefas executadas. Os questionários foram divididos em dois blocos:

   - Avaliação Quantitativa: respondidas individualmente e direcionadas para respostas numeradas em escalas de 1 a 5 indicando grau de satisfação, complexidade, etc. Os resultados dos questionários serão apresentados em gráficos e discutidos em detalhes nas seções seguintes.

– Avaliação Qualitativa: respondida pelas duplas, e composta de questões dissertativas abertas, permitindo ampla discussão a respeito dos funcionalidades da plataforma. Para interpretação dos resultados foram analisadas todas as repostadas e identificadas as recorrências de temas abordados, os quais serão comentados e discutidos nas próximas seções.

Os resultados da execução de cada grupo de procedimentos e respectivos questionários são apresentados nas Seções 5.3.2 e 5.3.3.

5.1 Estudos de caso

A validação em relação à independência de domínio de aplicação foi abordada realizando testes com bibliotecas digitais distintas e de domínios diferentes. Foram selecionados um *dataset* de avaliação de filmes freqüentemente citado na literatura [34,56] e um subconjunto da Biblioteca Digital da Unicamp como cenários de uso para a plataforma. As seções seguintes abordarão, em maiores detalhes, os testes e procedimentos executados.

5.1.1 Movielens

Em [26] é proposto um mecanismo de recomedação para filmes, baseado em técnicas colaborativas. Como base de experimentos foi utilizado um conjunto de dados de 100.000 *ratings* para 1.682 filmes, avaliados por 943 usuários. Esse conjunto de dados foi posteriormente disponibilizado na internet, através do site do Grupo de Pesquisas GroupLens (http://www.grouplens.org). Encontram-se na literatura diversos artigos que se utilizarm dessa base de dados para experimentos [34, 56].

Todavia esse conjunto de dados é bastante específico para técnicas colaborativas, dado que não possui conteúdo sobre os itens apresentados. Foi construída então, uma aplicação responsável por acessar o site MovieDB (http://us.imdb.com) e recuperar, para cada filme a sua respectiva sinopse.

Dessa forma, esse conjunto de dados ficou satisfatoriamente completo para os primeiros experimentos da plataforma, dispondo de informações para as técnicas colaborativas, baseadas em conteúdo e híbridas. Foram executados com sucesso testes de consulta e requisições de recomendação de todas as técnicas.

A Figura 5.1 ilustra esses testes: a parte (A) mostra a tela inicial do protótipo durante uma requisição de recomendação ao *engine* colaborativo ICMultilensRecommender; a parte (B) mostra os detalhes de uma consulta realizada ao filme “Batman Forever”; por fim a parte (C) exibe os itens recomendados para esse item, como “Die Hard: With a Vengeance” e “First Knight”.

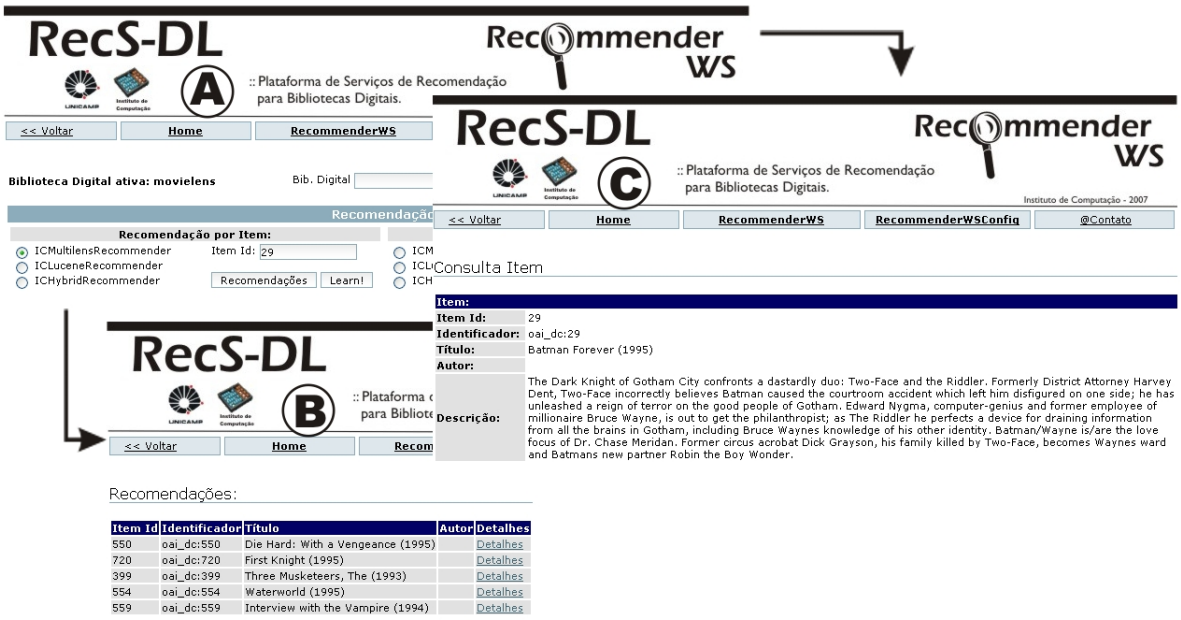


Figura 5.1: Recomendações da Plataforma RecS-DL ao *recordset* Movielens.

5.1.2 Biblioteca Digital da Unicamp

A Biblioteca Digital da Unicamp, foi oficialmente instituída em agosto de 2001, por meio da portaria GR-85, que trata da sua estruturação. O projeto foi efetivamente implantado no segundo semestre de 2002 [73], utilizando o sistema Nou-Rau [1], iniciativa de software livre da universidade. Desde então, consolidou-se a disponibilização do conteúdo da Biblioteca Digital da Unicamp à comunidade interna e externa, nacional e internacional, provendo um mecanismo de difusão de informação. Segundo dados obtidos até julho de 2006 [73], foi ultrapassado o número de 1 milhão de *downloads* de teses.

Dada a grande quantidade de informações e a demanda por novas funcionalidades [73], a Biblioteca Digital da Unicamp apresentou-se como um excelente cenário de uso para a plataforma de recomendação proposta. Foi selecionado para os experimentos um subconjunto contendo 6.760 itens (entre teses, dissertações, relatórios ténicos e outros) e 115.568 registros de *downloads* efetuados por 56.524 usuários distintos.

A plataforma de recomendação foi configurada em uma base de dados que continha este subconjunto da biblioteca. A aplicação web do protótipo foi utilizada para realizar os experimentos. Foram realizados testes de consulta e requisições de recomendação utilizando dados reais da Bibliteca Digital da Unicamp. A Figura 5.2 ilustra esses testes: a parte (A) mostra a tela inicial do protótipo durante uma requisição de recomendação ao

engine ICLuceneRecommender; a parte (B) mostra os detalhes de uma consulta realizada ao título "Ferramentas para comparação genomica"; por fim a parte (C) exibe os itens recomendados para esse item, como "Um algoritmo para comparação sintatica de genomas baseado na complexidade condicional de Kolmogorov", "Bioinformatica de projetos genoma de bacterias", entre outros. Como pode ser observado, há uma relação semântica entre as teses/dissertações recomendadas.

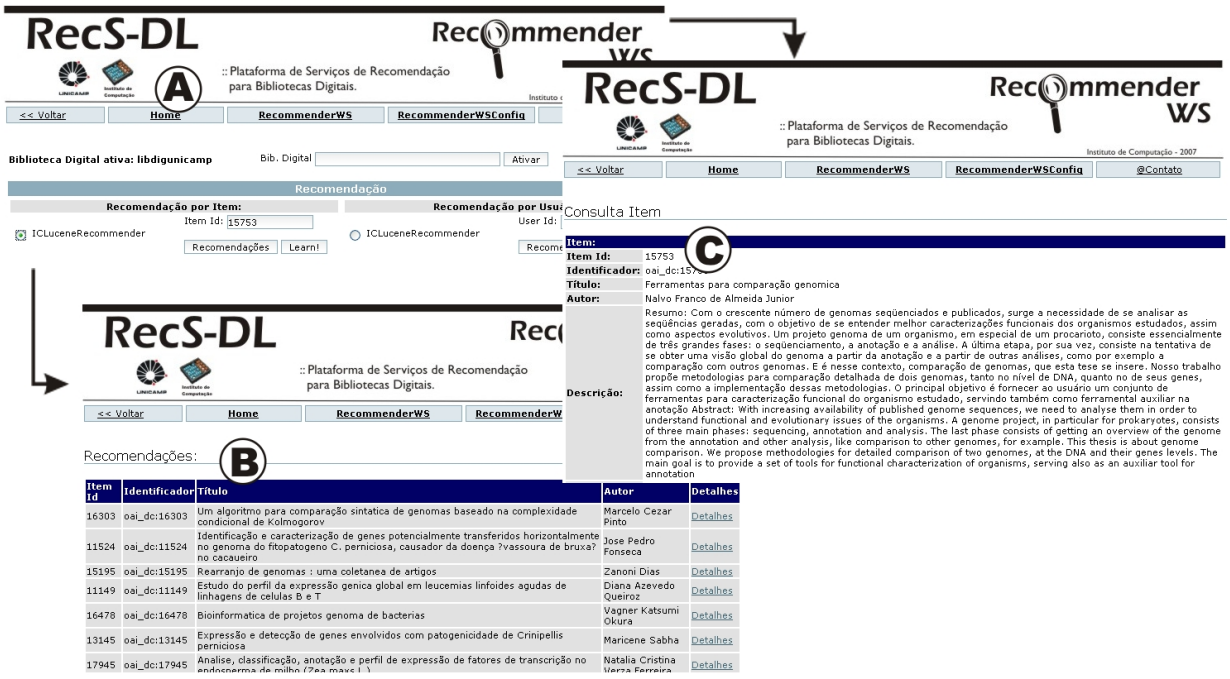


Figura 5.2: Recomendações da Plataforma RecS-DL à Biblioteca Digital da Unicamp.

5.2 Biblioteca Digital da Unicamp

Os experimentos realizados nessa etapa consistem na incorporação de funcionalidades de recomendação à Biblioteca Digital da Unicamp utilizando a Plataforma RecS-DL.

Diferentemente do estudo de caso apresentado na seção anterior, nesta seção de experimentos serão discutidos aspectos que vão além da utilização de um subconjunto do dados. O objetivo consiste em utilizar todo o banco de dados e integrar a Plataforma RecS-DL ao *software* Nou-Rau [1], que gerencia a Biblioteca Digtal da Unicamp.

A primeira etapa do experimento consistiu na criação de um ambiente de simulação da Biblioteca Digital da Unicamp. Foi utilizado um microcomputador Pentium IV, 3.0GHz,

2 GB de memória RAM DDR2 533 e HD de 300 GB 7200rpm. Esse microcomputador recebeu uma cópia fiel daquele que hospedava em produção a Biblioteca Digital da Unicamp em Julho/2007, com sistema operacional Linux Debian. Ao término deste procedimento, o *software* Nou-Rau e respectivas dependências e *softwares* servidores (Apache, PHP, PostgreSQL) estavam instalados e operantes.

A próxima etapa do experimento consistiu na instalação da Plataforma RecS-DL e dos *softwares* que são pré-requisito para o seu funcionamento. Todo o processo de instalação e posterior configuração da Plataforma RecS-DL foi realizado manualmente, dado que a ferramenta de instalação da plataforma ainda não é portável para Sistema Operacional Linux. Os procedimentos executados durante a instalação foram:

1. Instalação do J2SE.
2. Instalação do software Apache Ant.
3. Instalação do servidor Apache Tomcat.
4. Cópia da Plataforma RecS-DL.
5. Cópia da Aplicação PHP Cliente.

Concluída a instalação de todos os softwares pré-requisitos da plataforma, iniciou-se o procedimento de configuração da plataforma. Foram executados os seguintes procedimentos:

1. Configuração de Arquivos: arquivos de configuração da plataforma (.properties) foram editados e configurados para os caminhos onde a plataforma foi instalada.
2. Compilação e Instalação: foi realizado o procedimento de empacotamente (*build*) e instalação da plataforma no servidor de aplicações (*deploy*).
3. Engine de Recomendação: o engine de recomendação por conteúdo ICLuceneRecommender foi instalado na Plataforma RecS-DL.

Completa a configuração da Plataforma RecS-DL, o último conjunto de procedimentos a ser executado, consiste na integração da plataforma à Biblioteca Digital da Unicamp. As tarefas executadas foram:

1. Configuração da Biblioteca Digital: configuração da conexão do banco de dados do *software* Nou-Rau na Plataforma RecS-DL.
2. Criação de Views: criação das visões (*views*) definidas pela plataforma.

3. Atualização de Modelos: atualização dos índices do *engine* ICLuceneRecommender.

4. Integração do Software Nou-Rau: alteração nos fontes do *software* Nou-Rau permitindo que na consulta de cada item seja exibida a opção de solicitar recomendações à Plataforma RecS-DL.

Os problemas encontrados durante o processo de instalação são relacionados a seguir. Houve uma incompatibilidade do conjunto de carateres (*charset*) utilizado pelo banco de dados. A solução adotada foi a conversão para o *charset* padrão UTF8. Outro problema enfrentado consistiu na incompatibilidade de versões da Aplicação Cliente da plataforma, preparada para PHP versão 5, e o servidor PHP versão 4, previamente instalado. A solução foi realizar adaptações tornando a aplicação compatível também com o PHP versão 4.

A Figura 5.3 ilustra uma tela do *software* Nou-Rau exibindo uma consulta a uma tese da Biblioteca Digital da Unicamp, na parte (A). A seta ilustra a integração com a Plataforma RecS-DL, na parte (B), responsável por gerar recomendações para o item consultado.

5.3 Experimentos com potenciais usuários

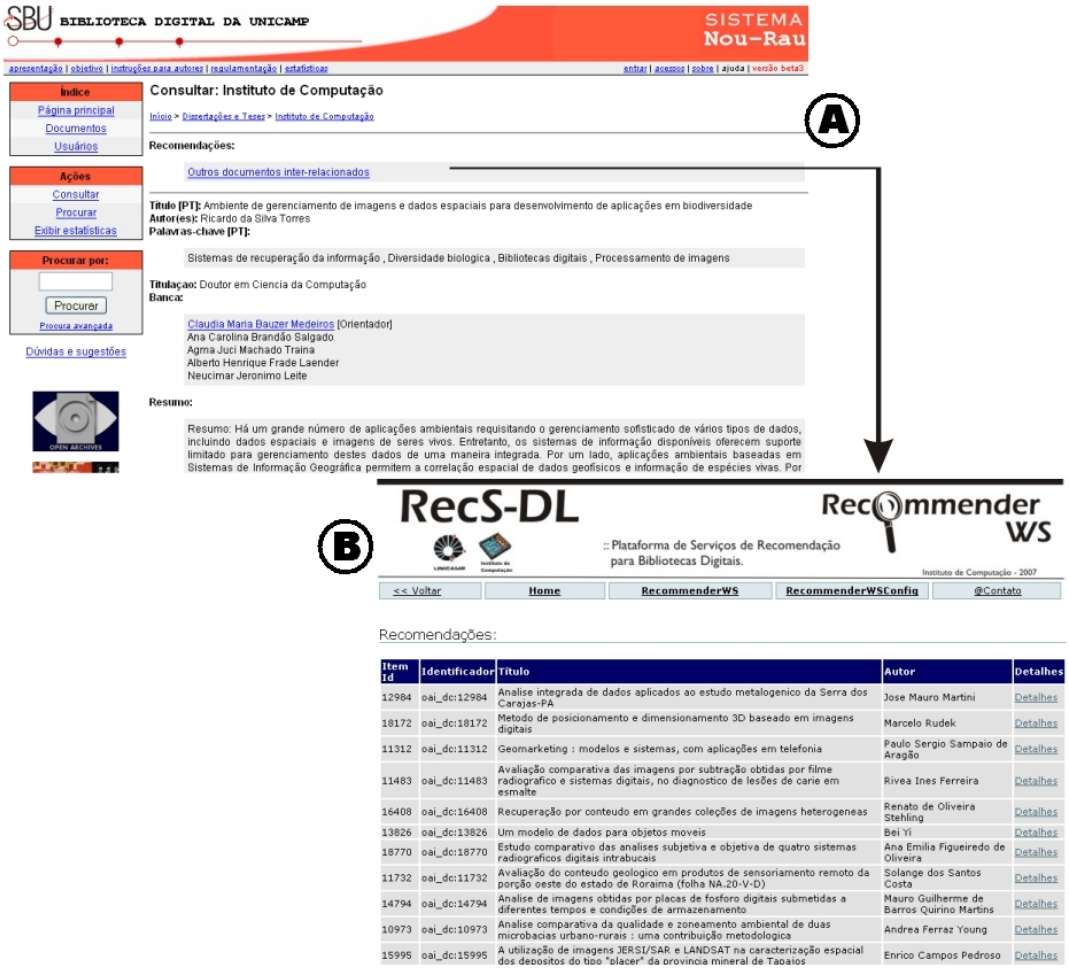
O principal público alvo da Plataforma RecS-DL consiste em administradores de bibliotecas digitais, usuários que podem adicionar serviços de recomendação às bibliotecas digitais gerenciadas utilizando a Plataforma RecS-DL. Assim, participaram dos experimentos estudantes de graduação e pós-graduação do Instituto de Computação da Unicamp, cursando a disciplina de Bibliotecas Digitais. Esses estudantes foram identificados como futuros potenciais administradores de bibliotecas digitais.

Os estudantes que participaram dos experimentos, divididos em grupos, executaram no início da disciplina a tarefa de criar uma biblioteca digital contendo Relatórios Técnicos do Instituto de Computação da Unicamp. Essa biblioteca digital foi posteriormente utilizada em alguns dos experimentos realizados.

Para os experimentos foram utilizadas duas versões da Plataforma RecS-DL. A primeira versão foi utilizada no primeiro experimento. Algumas melhorias e correções foram incluídas e uma segunda versão foi utilizada no segundo experimento.

5.3.1 Perfis dos Usuários

Foram construídos questionários com o objetivo de traçar um perfil sobre os usuários participantes dos experimentos. As questões procuraram levantar tanto informações acadêmicas, indagando a formação dos participantes, quanto informações técnicas, questio-



Item Id	Identificador	Título	Autor	Detalhes
12984	oai_dc:12984	Analise integrada de dados aplicados ao estudo metalogenico da Serra dos Carajas-PA	Jose Mauro Martini	Detalhes
18172	oai_dc:18172	Metodo de posicionamento e dimensionamento 3D baseado em imagens digitais	Marcelo Rudek	Detalhes
11312	oai_dc:11312	Geomarketing : modelos e sistemas, com aplicações em telefonia	Paulo Sergio Sampaio de Aragão	Detalhes
11483	oai_dc:11483	Avaliação comparativa das imagens por subtração obtidas por filme radiografico e sistemas digitais, no diagnostico de lesões de carie em esmalte	Rivea Ines Ferreira	Detalhes
16408	oai_dc:16408	Recuperação por conteudo em grandes coleções de imagens heterogeneas	Renato de Oliveira Stehling	Detalhes
13826	oai_dc:13826	Um modelo de dados para objetos moveis	Bei Yi	Detalhes
18770	oai_dc:18770	Estudo comparativo das analises subjetiva e objetiva de quatro sistemas radiograficos digitais intrabucais	Ana Emilia Figueiredo de Oliveira	Detalhes
11732	oai_dc:11732	Avaliação do conteudo geologico em produtos de sensoriamento remoto da porção oeste do estado de Roraima (folha NA.20-V-D)	Solange dos Santos Costa	Detalhes
14794	oai_dc:14794	Analise de imagens obtidas por placas de fosforo digitais submetidas a diferentes tempos e condições de armazenamento	Mauro Guilherme de Barros Quirino Martins	Detalhes
10973	oai_dc:10973	Analise comparativa da qualidade e zoneamento ambiental de duas microbacias urbano-rurais : uma contribuição metodologica	Andrea Ferraz Young	Detalhes
15995	oai_dc:15995	A utilização de imagens JERSI/SAR e LANDSAT na caracterização espacial dos depositos do tipo "placer" da provincia mineral de Tapajos	Enrico Campos Pedroso	Detalhes

Figura 5.3: *Software* Nou-Rau integrado à Plataforma RecS-DL.

nando sobre o nível de conhecimento dos participantes em diversas tecnologias. Todas as tecnologias questionadas estão relacionadas à Plataforma RecS-DL.

O questionário construído pode ser visto no Anexo B. Serão apresentados a seguir os gráficos com os resultados obtidos.

A Figura 5.4 apresenta o nível de formação dos participantes. Pode-se verificar que a aproximadamente metade, totalizando 47% dos participantes possuem nível superior. A outra metade divide-se entre alunos de graduação, com 27%, e mestrandos e doutorandos, com outros 27%. Apesar da diversidade de nível de formação, o perfil geral dos participantes foi considerado adequado, já que a grande maioria possui graduação completa ou está em nível de pós-graduação.

A área de formação dos participantes é dividida conforme os resultados apresentados

na Figura 5.5. Como se pode verificar, a maior parte dos participantes, totalizando 62%, são formados em "Ciência da Computação" ou "Engenharia da Computação".

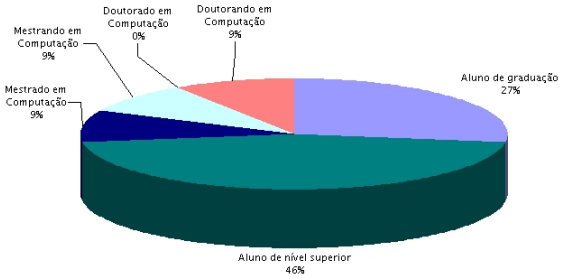


Figura 5.4: Formação dos participantes.

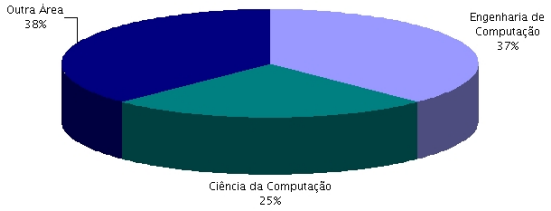


Figura 5.5: Área de formação dos participantes.

As Figuras 5.6, 5.7, e 5.8 apresentam os resultados do nível de conhecimento dos participantes em relação à tecnologia de Serviços Web. Embora 54% dos participantes tenham classificado como "Satisfatório" o nível de conhecimento, apenas 27% já haviam utilizado e apenas 9% já tinham implementado um Serviço Web.

De maneira geral, o nível de conhecimento na tecnologia é pequeno e pode ter influenciado os resultados dos experimentos. Deve-se considerar que para a maior parte dos participantes foi a primeira vez a utilizar um Serviço Web.

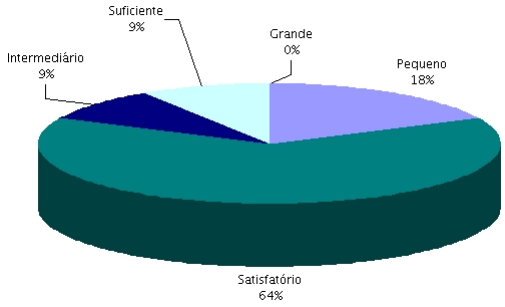


Figura 5.6: Conhecimento sobre Serviços Web.

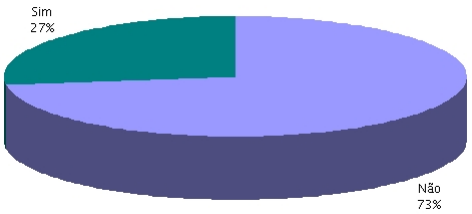


Figura 5.7: Utilização de Serviços Web.

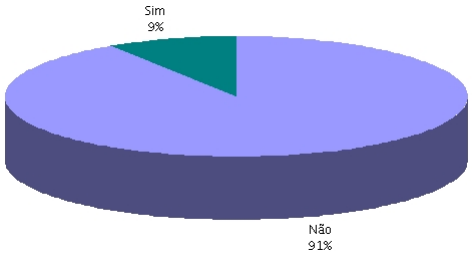


Figura 5.8: Implementação de Serviços Web.

O nível de conhecimento em banco de dados também foi questionado. Pôde-se verificar nos resultados apresentados na Figura 5.9 que 91% dos participantes já tinha implementado aplicações utilizando banco de dados. A Figura 5.10 apresenta os banco de dados mais utilizados.

As Figuras 5.11 e 5.12 apresentam os resultados do nível de conhecimento dos participantes sobre visões (*views*) de banco de dados. 64% dos participantes classificaram como "Pequeno" ou "Intermediário" o nível de conhecimento em visões e apenas 36% dos participantes já implementaram aplicações utilizando visões.

Analisando os resultados pode-se concluir que, embora os participantes tenham um bom nível de conhecimento em banco de dados, o mesmo não ocorre quanto ao uso de visões de banco de dados.

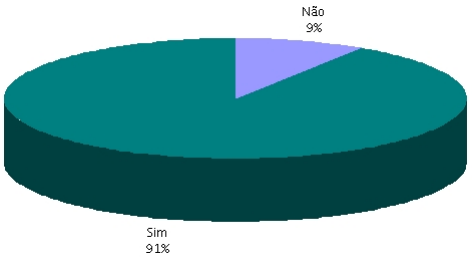


Figura 5.9: Implementação utilizando SGBDs.

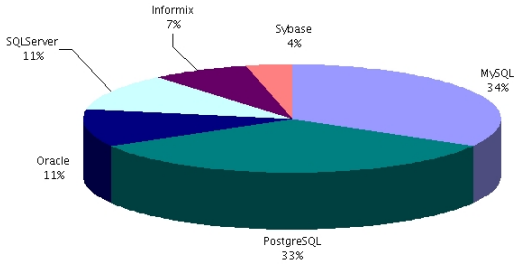


Figura 5.10: SGBDs já utilizados.

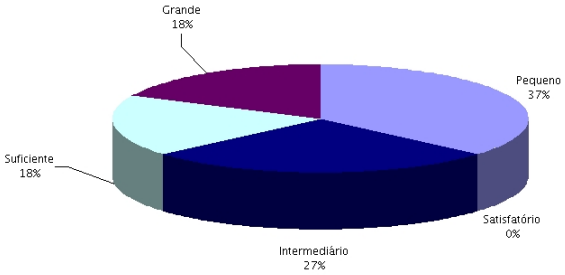


Figura 5.11: Conhecimento sobre *views*.

Figura 5.12: Implementação utilizando *views*.

Por fim, os participantes foram questionados acerca da utilização de serviços de recomendação. Apenas 18% dos particpantes já tinham utilizado um serviço de recomendação.

5.3.2 Experimento I: Instalação da plataforma

O objetivo deste experimento é avaliar algumas hipóteses em relação à instalação da Plataforma RecS-DL. A primeira delas consiste em certificar-se de que a instalação é um procedimento passível de execução por usuários potenciais, sem a necessidade de conhecimentos prévios sobre a plataforma. O experimento procurou avaliar também a

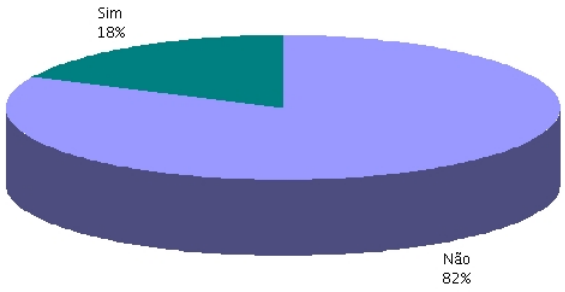


Figura 5.13: Utilização de serviços de recomendação.

hipótese de que a ferramenta de instalação desenvolvida auxilia significativamente no procedimento de instalação.

As tarefas descritas para execução do experimento consistem nos procedimentos necessários para que a Plataforma esteja instalada e efetivamente em funcionamento. A descrição completa das atividades executadas neste experimento estão apresentadas no formulário do Anexo C.

Questionário Quantitativo

Os resultados quantitativos dos experimentos são apresentados a seguir, seguidos de análise das justificativas apresentadas nos relatórios e questionários qualitativos. A Figura 5.14 apresenta os resultados da avaliação do procedimento de instalação quanto à complexidade. Como já era previsto, dada a necessidade de instalação de vários softwares servidores, a avaliação da instalação teve resultados medianos. Enquanto 37% dos participantes classificaram como “Simples”, 43% classificaram como “Complexo” ou “Muito Complexo”.

Outra hipótese a ser verificada era a de que a ferramenta de instalação desenvolvida poderia facilitar o procedimento de instalação. A Figura 5.15 apresenta um gráfico que exibe resultados bastante positivos da avaliação da ferramenta de instalação. A avaliação foi realizada em relação à simplicidade na manipulação da ferramenta, onde 80% dos participantes classificaram como “Simples” ou “Muito Simples” . Tais resultados apresentam bons indícios de que o objetivo de simplificar o processo de instalação foi alcançado com sucesso.

Além da simplicidade da ferramenta de instalação, pretendia-se avaliar também a efetividade da ferramenta. A Figura 5.16 apresenta os resultados da avaliação de eficácia, ou seja, se a ferramenta atende às necessidades. Embora os resultados gerais da avaliação sejam bons, com classificações positivas de aproximadamente 67% dos participantes, ainda

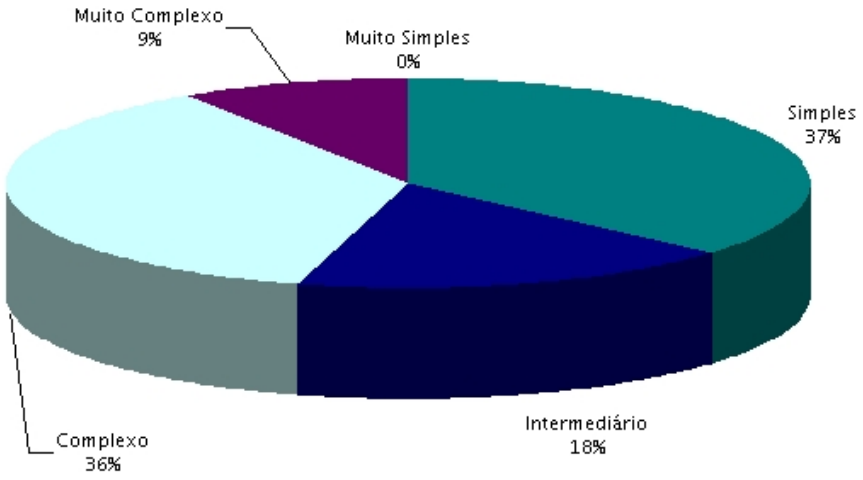


Figura 5.14: Avaliação do procedimento geral de instalação.

há 9% e 27% dos participantes que avaliaram que a ferramenta "Não Atende" ou "Atende Parcialmente". As avaliações negativas foram atribuídas aos *bugs* de implementação encontrados na primeira versão da ferramenta, que serão discutidas a seguir, na análise dos questionários qualitativos.

Vale lembrar também que, independente dos resultados das avaliações, todos os participantes conseguiram executar completamente os procedimentos estipulados pelo experimento. Esse fato, por si só, já confirma a possibilidade de execução da instalação da plataforma por potenciais usuários.

Questionário Qualitativo

Vários participantes apontaram como aspecto positivo a simplicidade da instalação utilizando a ferramenta oferecida pela plataforma, composta de poucos procedimentos e isolando o usuário de detalhes específicos dos servidores utilizados. A intuitividade e simplicidade da *interface* da ferramenta de instalação também foi citada. A integração dos *softwares* servidores em um único instalador foi um aspecto positivo recorrentemente citado.

Quanto ao tempo e desempenho da ferramenta de instalação, as avaliações foram predominantemente positivas. Vários participantes classificaram como rápido o procedimento de instalação. A ferramenta de controle dos servidores criada também foi citada como aspecto positivo por quase todos os partipantes. Embora alguns participantes tenham relatado lentidão na execução de algumas tarefas, essa ferramenta possibilitou que

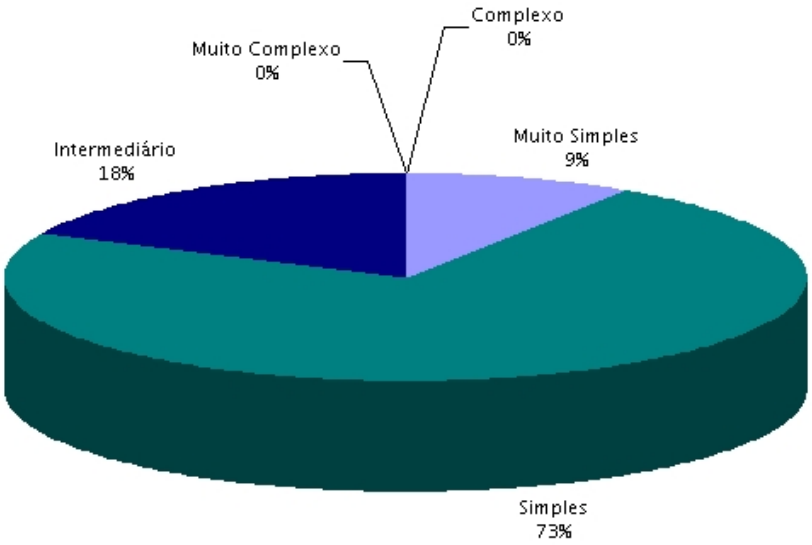


Figura 5.15: Avaliação da manipulação da ferramenta de instalação.

o procedimento de inicialização dos servidores fosse bastante facilitado. Em relação à documentação, o Guia Rápido foi citado como aspecto positivo, atingindo o objetivo de auxiliar o usuário durante a instalação.

Alguns *bugs* de implementação foram identificados durante os experimentos. A ferramenta de instalação apresentava, em sua primeira versão, problemas para instalação em pastas cujos nomes eram extensos. A primeira versão também não apresentava um relatório de possíveis erros para o usuário ao término da instalação, resultando em sucesso em todas as situações. A ferramenta de controle dos *softwares* servidores não testava se havia outros servidores ativos antes da execução. Tais problemas demandaram correções e geraram novas funcionalidades para as ferramenta de instalação e controle. As correções foram efetuadas e os participantes puderam experimentar a nova versão nos experimentos seguintes. Processo análogo ocorreu com a documentação, onde pequenos erros, geralmente relacionados a aspectos de redação, foram apontados durante os experimentos e posteriormente corrigidos.

Várias possíveis melhorias também foram apontadas nas respostas dos questionários. Certamente, várias delas constituem importantes trabalhos futuros. Entre os aspectos negativos e futuras melhorias mais recorrentemente apontados estão a necessidade de um desinstalador, e a portabilidade da ferramenta de instalação para ambiente Linux. Outros aspectos negativos menos relevantes citados foram a evolução não linear da barra de progresso e a falta de informações para resolução de exceções. A barra de progresso

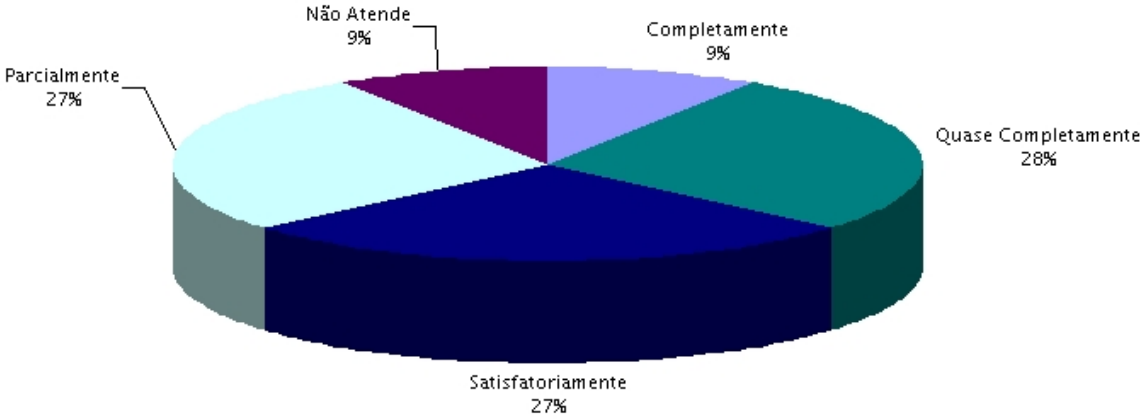


Figura 5.16: Avaliação: A ferramenta de instalação atende às necessidades?

citada é exibida durante a cópia dos arquivos e execução de procedimentos de configuração. Em relação à documentação, vários participantes apontaram a falta de documentação sobre a ocorrência e tratamento de erros. Uma sugestão apresentada para resolução deste problema, que certamente constitui um importante trabalho futuro é um documento de FAQs (*Frequently Asked Questions*).

5.3.3 Experimento II: Aquisição de dados e utilização da plataforma

Este experimento tem como objetivo avaliar a Plataforma RecS-DL em relação à importação de dados e a integração a outros softwares de bibliotecas digitais. A primeira hipótese a ser verificada consiste na possibilidade da importação de dados de bibliotecas digitais pela Plataforma RecS-DL. A segunda hipótese tem como objetivo verificar a possibilidade de integração dessas bibliotecas à plataforma, de modo a oferecer serviços de recomendação.

As bibliotecas digitais importadas são compostas por Relatórios Técnicos do Instituto de Computação da Unicamp, utilizando os softwares Greenstone [74], Fedora [50] e D-Space [69]. A descrição completa das atividades definidas pelos experimentos estão apresentadas no formulário do Anexo D.

Questionário Quantitativo

Os resultados dos experimentos são apresentados a seguir. A primeira avaliação, de importação de dados, que é executada pelo Módulo de Aquisição da plataforma é apresentada

na Figura 5.17. Os resultados são ligeiramente positivos, apresentando 60% das avaliações entre "Quase satisfatórios" e "Satisfatórios". Entretanto, 40% das avaliações entre intermediário e negativo indicam possíveis problemas neste módulo.

Uma provável causa do índice de insatisfação apresentado pelo Módulo de Aquisição, pode ser encontrado nas avaliações das formas de importação de dados, apresentadas a seguir. O participantes avaliaram o nível de complexidade das três formas de importação de dados: via OAI, visões de banco de dados e Serviços Web. Os resultados são apresentados nas Figuras 5.18, 5.19 e 5.20. Como é possível constatar nos gráficos, as três formas foram, em variados graus, classificadas como complexas.

Embora todos os participantes tenham concluído completamente o procedimento de importação de dados, objetivo principal do experimento, a avaliação quanto a complexidade foi predominantemente negativa. Tal avaliação pode ter sido causada por pouco conhecimento das tecnologias utilizadas pelos participantes, como pode ser verificado na Seção 5.3.1. Entretanto pode também indicar problemas quanto aos métodos criados. Assim, melhorias no Módulo de Aquisição, como as apontadas no questionário qualitativo apresentado a seguir, constituem trabalhos futuros relevantes.

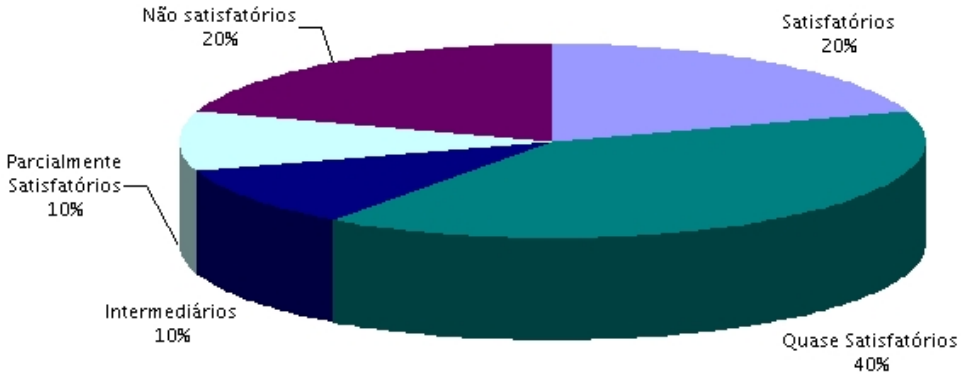


Figura 5.17: Avaliação dos resultados obtidos pelo Módulo de Aquisição.

A próxima avaliação refere-se ao procedimento de integração da Plataforma RecS-DL a outros *softwares* de bibliotecas digitais. Esse procedimento era o maior candidato a apresentar dificuldades, já que demanda esforços de criação ou alteração de aplicações capazes de interagir entre a Plataforma RecS-DL e o *software* escolhido.

Apesar das dificuldades, todos os participantes concluiram completamente as atividades dos experimentos. Tais resultados ratificam a hipótese de que a Plataforma RecS-DL pode oferecer um serviço de recomendação a ser incorporado em outros *softwares* de bibliotecas digitais. Os relatórios construídos pelos participantes apresentam telas onde os

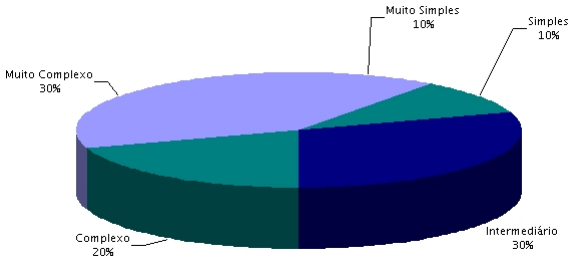


Figura 5.18: Avaliação do procedimento de importação de dados utilizando OAI.

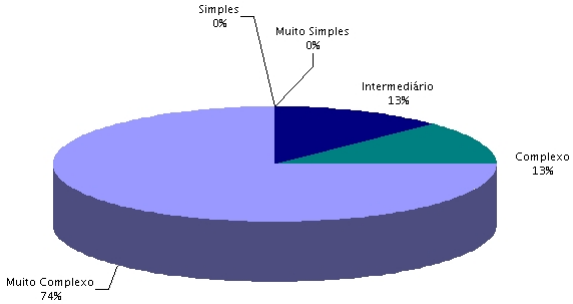


Figura 5.19: Avaliação do procedimento de importação de dados utilizando *Web-Services*.

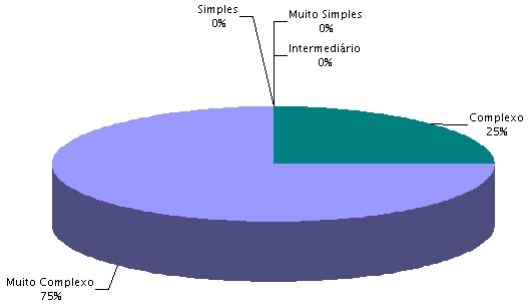


Figura 5.20: Avaliação do procedimento de importação de dados utilizando *views*.

softwares utilizados foram integrados à Plataforma RecS-DL. A Figura 5.21 exibe o *software* Greenstone, a Figura 5.22 o *software* Fedora, a Figura 5.23 o *software* DSpace, e por fim, a Figura 5.24 ilustra tela da Plataforma RecS-DL apresentando as recomendações em resposta à requisição do *software* DSpace.

Foi realizada também pelos participantes uma avaliação quanto a complexidade do procedimento de integração, cujos resultados estão apresentados na Figura 5.25. Dadas as dificuldades previstas para os procedimentos, podem-se considerar os resultados positivos, já que ainda assim 51% dos participantes classificaram as atividades de "Simples" ou "Muito Simples". A variação siginificativa dos resultados pode ser provavelmente atribuída à utilização de vários *softwares*, criando cenários e níveis de dificuldades distintos. Outras possibilidades serão discutidas na análise dos experimentos qualitativos.

Por fim, a última avaliação quantitativa realizada refere-se ao potencial de utilidade da Plataforma RecS-DL, como provedora de serviços de recomendação. Os participantes dos experimentos, após a instalação e utilização da plataforma realizaram uma avaliação do potencial de utilidade da plataforma. Os resultados, ilustrados na Figura 5.26, são muito positivos: 87% dos participantes classificaram a Plataforma RecS-DL como "Útil"

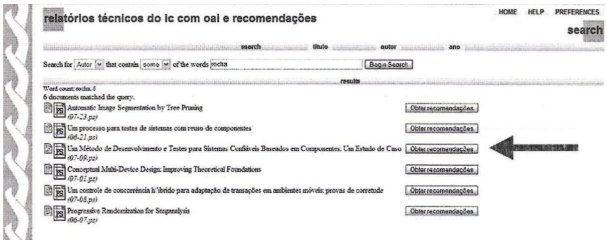


Figura 5.21: Integração com o *software* Greenstone.

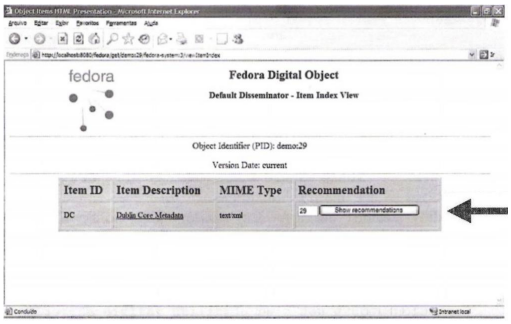


Figura 5.22: Integração com o *software* Fedora.

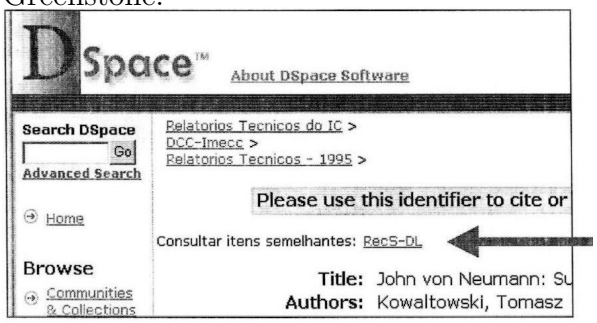


Figura 5.23: Integração com o *software* DSpace.

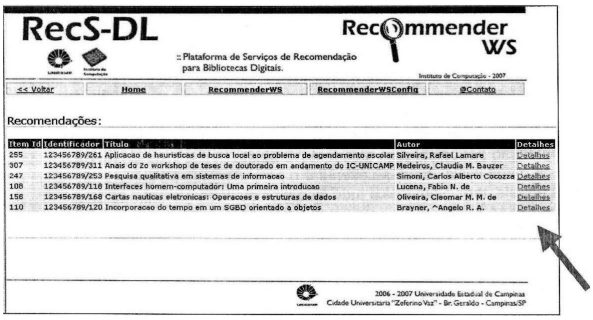


Figura 5.24: Plataforma RecS-DL integrada ao *software* DSpace.

ou "Muito útil". A avaliação apresenta bons indícios de que as funcionalidades oferecidas estão adequadas.

Questionário Qualitativo

Os relatórios analisados apontaram para o importante resultado de que todos os participantes concluíram as atividades propostas pelo experimento, configurando e integrando a Plataforma RecS-DL às bibliotecas digitais. Todavia, os procedimentos de importação de dados e integração a outros softwares de bibliotecas digitais tiveram mais pontos de dificuldades apontados do que o procedimento de instalação. De certa forma, isso era esperado, dado o maior nível de complexidade da tarefa proposta neste experimento. A seguir serão analisados os motivos apontados e as conclusões geradas.

O Módulo de Aquisição de Dados foi o que concentrou as maiores dificuldades encontradas. A importação de dados via protocolo OAI foi classificada como simples pela maioria dos participantes, e executada para os *softwares* DSpace, Greenstone e Fedora. Entretanto, vários problemas foram apontados. O tempo necessário para importação grandes bases de dados e o acompanhamento do processo foram as dificuldades mais cita-

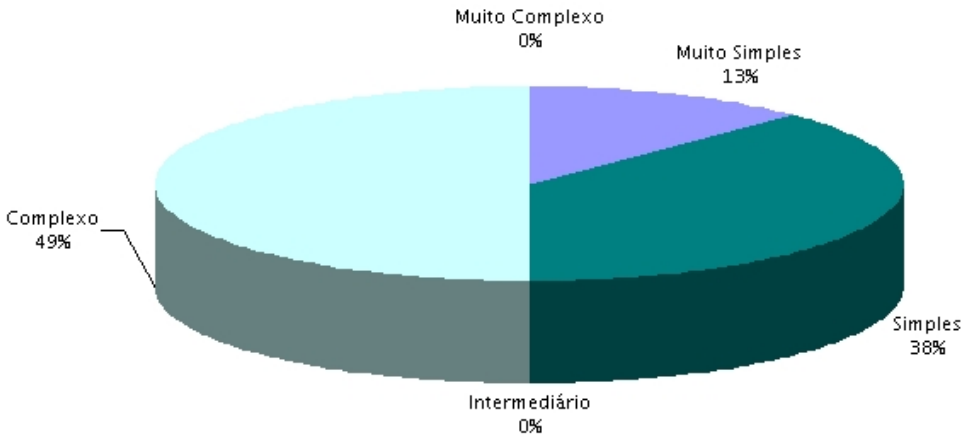


Figura 5.25: Avaliação do processo de integração da Plataforma RecS-DL a outros softwares de bibliotecas digitais.

das. Por se tratar de um procedimento que demanda muito tempo e utiliza *interface* Web, executa em segundo plano (*background*) no servidor para evitar *timeouts*. Executando em segundo plano, informações sobre o andamento da execução não são exibidas ao usuário. Métodos e uma *interface* que viabilizem a apresentação do estado da execução certamente constituem um trabalho futuro. A necessidade de maior utilização do identificador dos metadados OAI de cada item também foi uma necessidade apontada, já que a plataforma trabalha apenas com identificadores numéricos.

Outras dificuldades relatadas, de menor relevância, consistiram na sensibilidade deste módulo a maiúsculas e minúsculas, a impossibilidade de trabalhar com atributos de metadados com o mesmo nome (Autor="X", Autor="Y") e o não tratamento pela plataforma de parâmetros não obrigatórias do protocolo OAI. A documentação também foi um apontada como insuficiente em alguns casos. Asssim a documentação passou por correções, apresentando mais detalhes sobre as dificuldades relatadas.

A importação de dados utilizando *views* foi executada nos experimentos para o *software* DSpace. O software Greenstone não utiliza banco de dados relacional. Um aspecto positivo apontado para as *views* consistiu no acesso *on-line* pela Plataforma RecS-DL às bases da biblioteca digital, dispensando procedimentos periódicos de atualização. Aspectos negativos apontados consistiram na necessidade de conhecimento do modelo de dados do *software* da biblioteca digital. Os experimentos apontaram que, embora a plataforma já tivesse sido testada com os SGBDs MySQL e PostgreSQL, o funcionamento com o SGBD Oracle demandou duas pequenas correções: quanto ao uso da palavra-chave *date* que não pode ser utilizada como nome de atributo e o tipo *boolean*, que não é suportado

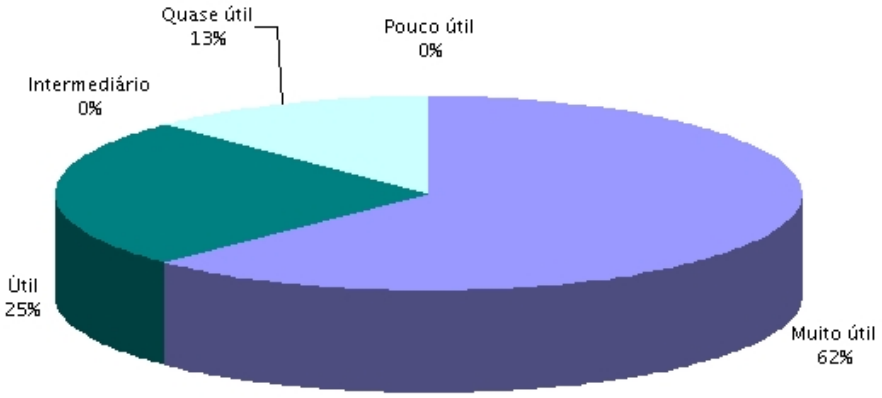


Figura 5.26: Avaliação da utilidade da Plataforma RecS-DL.

pelo Oracle.

A importação de dados utilizando Serviços Web foi executada por 4 participantes. A desvantagem apontada por esse método foi a necessidade de construção de programas para acesso aos Serviços Web e conhecimento técnico sobre as tecnologias associadas. Esse foi o motivo pelo qual os outros participantes não executaram esse procedimento. Entretanto, todos os participantes que executaram o procedimento relataram que as operações funcionaram como esperado e não apresentaram falhas.

O procedimento de integração com outros *softwares* de bibliotecas digitais foi executado por todos os participantes, utilizando requisições HTTP, por meio do método POST, à *interface* da aplicação cliente da plataforma. Como já havia ficado evidente nos experimentos quantitativos, praticamente todos os participantes classificaram como simples este procedimento. Nos relatórios foram incluídos os códigos fonte responsáveis pelas integrações, em todos os casos constituídos de poucas linhas de código.

De maneira mais geral, aspectos positivos apontados consistiram na existência de várias formas de importação de dados, possivelmente mais adaptáveis às necessidades diversas. Outro aspecto positivo apontado foi a existência de uma interface *Web* de configuração, permitindo a execução remota e dispensando edição de arquivos de configuração. Como aspectos negativos foram apontadas recorrentemente a *interface* da aplicação cliente e seu tratamento e mensagem de erros incompletas. A aplicação cliente foi inicialmente desenvolvida com o objetivo de testar os métodos dos Serviços Web da plataforma. Assim, requisitos de usabilidade não foram considerados, e constituem possíveis trabalhos futuros para a Plataforma RecS-DL como ferramenta destinada ao usuário final. Possíveis novas funcionalidades apontadas são métodos de alteração e exclusão de dados e bibliotecas

digitais. Algumas correções na documentação, como restrições de nomes para bibliotecas digitais, também foram sugeridas. Tais correções já foram efetuadas.

Outra observação recorrente nos experimentos é de que, embora as ferramentas disponibilizadas tenham facilitado vários procedimentos, conhecimentos de programação, banco de dados e *softwares* servidores (Apache, Tomcat, etc) ainda são indispensáveis para configuração da plataforma. A Seção 5.3.1 apresenta os resultados do questionário realizado entre os participantes visando identificar o nível de conhecimento sobre as tecnologias. A inclusão de documentação auxiliar na ferramenta também pode auxiliar nesse sentido.

Uma conclusão quase que consensual entre os participantes é de que a Plataforma RecS-DL representa uma ferramenta inovadora, de grande utilidade e funcional, haja vista a efetiva conclusão dos experimentos, mas que melhorias seriam muito importantes para oferecimento da Plataforma RecS-DL como produto final.

Capítulo 6

Conclusões

Este capítulo apresenta as considerações finais desse trabalho, discutindo as suas contribuições e trabalhos futuros.

6.1 Contribuições

O aumento do volume de informações nos sistemas de bibliotecas digitais torna cada vez mais difíceis as tarefas de localização e escolha do conteúdo desejado. Nesse contexto, técnicas de recomendação capazes de facilitar o trabalho de escolha e localização dos objetos digitais de interesse são de grande valia aos usuários.

Com o crescimento das bibliotecas digitais em quantidade e abrangência, uma plataforma capaz de oferecer serviços de recomendação para domínios diversos, independente de tecnologia e de técnicas de recomendação representa uma ferramenta bastante relevante. Todavia, a maioria das ferramentas de recomendação existentes são restritas ao domínio, técnicas de recomendação ou tecnologia.

Visando propor soluções para a interoperabilidade das ferramentas de recomendação, esta dissertação apresentou várias contribuições:

- **Estudo comparativo das ferramentas existentes:** abordando as vantagens e limitações das ferramentas descritas na literatura. Esse estudo também constituiu a base para a identificação dos requisitos desejáveis a uma plataforma de serviços de recomendação;

- **Projeto e implementação da Plataforma RecS-DL:** a plataforma de recomendação proposta unifica as vantagens das várias ferramentas descritas na literatura, integrando Serviços Web, *interfaces* formais, independência de domínio, técnicas de recomendação e linguagem das aplicações clientes. Outras contribuições importantes consistem no Serviço de Configuração oferecido e no modelo arquitetural da

plataforma, que se baseia no conceito de *engines*. Tal modelo permite que diversas técnicas de recomendação possam ser implementadas, instaladas, e inclusive avaliadas sob uma mesma interface, provida pela plataforma proposta;

- **Formalização da Plataforma RecS-DL:** outra contribuição deste trabalho consiste nas extensões propostas ao Arcabouço 5S. Foram propostas formalizações de conceitos relacionados a recomendação e *software*. Essas formalizações, por sua vez, foram utilizadas para a definição formal da Plataforma RecS-DL.

- **Validação:** os experimentos conduzidos puderam verificar várias hipóteses levantadas durante o projeto e implementação da plataforma. Os resultados obtidos demonstraram que as funcionalidades oferecidas pela Plataforma RecS-DL são úteis e efetivamente aplicáveis a bibliotegas digitais reais. Os experimentos comprovaram que os procedimentos de instalação, importação de dados e integração a outros softwares de bibliotecas digitais são factíveis por potenciais usuários sem experiência prévia com a plataforma. Os resultados obtidos mostraram avaliações predominantemente positivas pelos participantes. Os experimentos indicaram também correções de *software* e documentação, algumas já efetuadas, melhorando a qualidade da plataforma proposta.

- **Publicações:** foram publicados dois artigos [51, 52], descrevendo a Plataforma RecS-DL e os estudos de caso realizados.

6.2 Extensões e trabalhos futuros

Além das contribuições, várias oportunidades de trabalhos futuros foram identificadas. Os experimentos apontaram possibilidades de melhorias e novas funcionalidades na Plataforma RecS-DL, principalmente no Módulo de Aquisição. Apesar de efetivamente funcional, tais melhorias certamente aumentariam a usabilidade da plataforma, aproximando-a de um produto pronto para os usuários finais.

Outros possíveis trabalhos futuros para a Plataforma RecS-DL consistem na implementação de novos *engines*, utilizando outras técnicas de recomendação. A ampliação da validação dada à plataforma por meio da utilização por outras bibliotecas digitais reais também é um teste relevante para os requisitos de flexibilidade em relação à diversidade de domínios e tecnologias.

Em relação às novas funcionalidades há várias possibilidades de trabalhos futuros:

- ***Relevance Feedback*** (Realimentação de Relevância): métodos para inclusão de informações de *relevance feedback* à Plataforma RecS-DL. As técnicas de *relevance*

feedback permitem que os usuários classifiquem a qualidade dos resultados apresentados em um conjunto de recomendações, possibilitando o aprendizado do sistema.

- **XMLLog:** a importação de dados no formato XMLLog [25] permitiria que informações colaborativas (histórico de usuários) de bibliotecas digitais fossem facilmente importadas pela Plataforma RecS-DL.

- **Novos métodos:** os usuários participantes dos experimentos indicaram que a incorporação de métodos (edição, exclusão, etc) ao Módulo de Aquisição da plataforma é uma demanda de novas funcionalidades desejáveis, visando melhorar a usabilidade da plataforma.

- **Repositório único:** estudo de viabilidade de utilização de um único repositório de treinamento para todas as bibliotecas digitais clientes da Plataforma RecS-DL, tendo em vista a possibilidade de utilização de dados colaborativos de bibliotecas digitais distintas.

- **Serviços Web nas *interfaces* internas:** estudo para verificar a viabilidade e as possíveis vantagens (maior flexibilidade) e desvantagens (*overload*) de utilização de Serviços Web como *interface* entre os módulos da Plataforma RecS-DL.

  A Plataforma RecS-DL também abre caminhos para outras abordagens e novas possíveis aplicações. Uma possível aplicação, diferente da original, seria a utilização da plataforma como meio de comparação entre técnicas e ferramentas de recomendação. Tal procedimento seria certamente mais simples utilizando a plataforma, dado que estariam todas as ferramentas operando sob as mesmas *interfaces*, definidas pela Plataforma RecS-DL.

Apêndice A

Banco de dados da Plataforma RecS-DL

Este anexo tem como objetivo apresentar os comandos DDL (*Data Definition Language*) de criação do banco de dados utilizado pela Plataforma RecS-DL.

CREATE TABLE item (itemid serial NOT NULL, identifier varchar(250) NOT NULL, title text, creator text, description text, base64bincontents text, filename text, date text, subject text, language text, format text, type text, contributor text, coverage text, publisher text, relation text, rights text, source text, textcontents text, CONSTRAINT itempk PRIMARY KEY (identifier), CONSTRAINT UniqueItemId UNIQUE (itemid));

CREATE TABLE rating (userid integer NOT NULL DEFAULT 0, itemid integer NOT NULL DEFAULT 0, rating double precision NOT NULL DEFAULT 0, ratingdate text, CONSTRAINT uipair PRIMARY KEY (userid, itemid));

CREATE TABLE userdb (userid integer NOT NULL DEFAULT 0, username text, name text, mail text, age integer DEFAULT 0, profission text, education text, region text, country text, interests text, CONSTRAINT userpk PRIMARY KEY (userid));

CREATE TABLE engine (classname text NOT NULL, alias varchar(100) NOT NULL, isdefault boolean, CONSTRAINT enginepk PRIMARY KEY (alias));

```
CREATE TABLE parameter (classname varchar(200) NOT NULL, paramname varchar(200) NOT NULL, paramtype text, value text, CONSTRAINT parameterpk PRIMARY KEY (classname, paramname));
```

Apêndice B

Questionário

Qual a sua formação?

a) Aluno de graduação (Engenharia de Computação, Ciência da Computação, Outra área)

b) Aluno de nível superior (Engenharia de Computação, Ciência da Computação, Outra área)

c) Mestrado em Computação

d) Mestrando em Computação

e) Doutorado em Computação

f) Doutorando em Computação

Como você avalia o seu conhecimento sobre Web services?

(1) Pequeno (2) (3) (4) (5) Grande

Você já usou Web Services?

(1) Não (2) Sim

Você já implementou Web Services?

(1) Não (2) Sim

Você já implementou sistemas de informação que usaram SGBDs?

(1) Não (2) Sim

Quais SGBDs você já utilizou?

( ) MySQL ( )PostegreSQL ( )Oracle ( )SQLServer ( ) Outros:

Como você avalia o seu conhecimento sobre Visões (Views) ?

(1) Pequeno (2) (3) (4) (5) Grande

Você já implementou sistemas de informação que usaram Visões?

(1) Não (2) Sim

Você já usou um serviço de recomendação anteriormente?

(1) Não (2) Sim

Qual (quais):

Apêndice C

Experimento I

Este projeto tem por objetivo validar a plataforma de recomendação RecS-DL, recentemente proposta pelo Instituto de Computação da Unicamp.

Instalação da plataforma RecS-DL

Descrição: o objetivo deste experimento é a realização de todos os procedimentos necessários para que a Plataforma RecS-DL esteja efetivamente operacional.

Esse conjunto de procedimentos inclui:

(i) a instalação de todos os softwares servidores necessários;

(ii) a configuração dos softwares servidores

(iii) a instalação e configuração da plataforma da Plataforma RecS-DL.

(iv) a instalação e configuração da interface web da plataforma.

A execução desses procedimentos deve ser realizada utilizando-se a ferramenta de instalação da plataforma. A documentação disponibilizada no site da Plataforma deve ser utilizada como primeiro material de consulta. O site da plataforma pode ser encontrado na página http://www.students.ic.unicamp.br/ra050269/recsdl.

Questionário qualitativo (respondido em grupo):

a) Quais os aspectos positivos e negativos em relação ao procedimento geral de instalação da plataforma?

b) Quais os aspectos positivos e negativos em relação à ferramenta de instalação da plataforma?

c) Quais os aspectos positivos e negativos em relação à ferramenta de controle da plataforma (start/stop dos servidores)?

d) Sinta-se a vontade para fazer comentários sobre a experiência de instalação da plataforma.

Questionário quantitativo (respondido individualmente):

e) Em relação à manipulação da ferramenta de instalação, pode-se dizer que é:

(5) Muito simples (4) Simples (3) Intermediário (2) Complexo (1) Muito complexo

f) Em relação aos procedimentos necessários para a instalação da Plataforma RecS-DL, podemos dizer que a ferramenta de instalação atende às necessidades:

(5) Completamente (4) Quase completamente (3) Satisfatoriamente (2) Parcialmente (1) Não atende

g) Em relação ao nível de dificuldade do procedimento geral de instalação, pode dizer que é:

(5) Muito simples (4) Simples (3) Intermediário (2) Complexo (1) Muito complexo

Apêndice D

Experimento II

Este projeto objetiva continuar a avaliação da ferramenta RecS-DL.

Importação de Dados

Descrição: Dada a biblioteca digital de relatórios técnicos criada no Projeto 1, realize os procedimentos necessários para que os dados desta biblioteca sejam importados pela plataforma RecS-DL. Essa tarefa pode ser realizada de várias formas diferentes:

- Utilizando o módulo de importação OAI;
- Por meio dos métodos de aquisição/inserção de dados implementados por meio de Web services;
- Criando visões (views) nas bibliotecas digitais originais.

O grupo deve utilizar (testar) todos os três métodos de importação de dados. Caso o sistema utilizado para a criação da biblioteca digital não possibilite o uso de alguns dos métodos, isso deve ser documentado.

Questionário qualitativo (respondido em grupo):

a) Quais os aspectos positivos e negativos em relação à importação de metadados pelo módulo OAI?

b) Quais os aspectos positivos e negativos em relação à importação de metadados por meio de Web services?

c) Quais os aspectos positivos e negativos em relação à importação de metadados por meio de visões (views)?

d) Sinta-se a vontade para fazer comentários sobre a experiência de importação de dados pela plataforma.

Questionário quantitativo (respondido individualmente):

e) Em relação ao procedimento de importação de metadados OAI pela plataforma RecSDL, pode-se dizer que é:

(5) Muito simples (4) Simples (3) Intermediário (2) Complexo (1) Muito complexo

f) Em relação ao procedimento de importação de dados por meio de Web Services, podese dizer que são:

(5) Completamente (4) Quase completamente (3) Satisfatoriamente (2) Parcialmente (1) Não atende

g) Em relação ao procedimento de importação de dados por meio de visões (views), podese dizer que é:

(5) Muito simples (4) Simples (3) Intermediário (2) Complexo (1) Muito complexo

h) Em relação aos resultados obtidos pelo módulo de aquisição/importação de dados, pode-se dizer que são:

(5) Satisfatórios (4) (3) (2) (1) Não satisfatórios

Utilização da Plataforma

Descrição: Dada a biblioteca digital de relatórios técnicos implementada no Projeto 1, e cujos dados já foram importados para a plataforma RecS-DL, integre a plataforma ao software que opera a biblioteca digital (DSpace, Greenstone, etc), criando

uma interface para solicitação e visualização de recomendações (vale lembrar que antes da solicitação de recomendações, deve-se treinar o engine de recomendação). Utilize os engines de recomendação disponíveis no site que descreve a ferramenta.

Um possível ponto de partida para a realização do procedimento de integração seria criar um link na interface do software da biblioteca digital para a interface web da Plataforma RecS-DL. Esse link pode enviar como um parâmetro http/post o identificador do item que se deseja obter recomendações. Essa alternativa deve demandar pequenas alterações em ambas as interfaces.

Outra alternativa possível consiste na implementação de clientes dos Web Services da Plataforma RecS-DL, implementados na interfaces do software da biblioteca digital.

Questionário qualitativo (respondido em grupo):

i) Quais os aspectos positivos e negativos em relação ao procedimento de integração da plataforma RecS-DL ao sistema de biblioteca digital utilizado pelo seu grupo?

j) Quais os aspectos positivos e negativos em relação ao procedimento de configuração da plataforma RecS-DL?

k) Você acredita que a plataforma RecS-DL pode ser útil como nova funcionalidade de bibliotecas digitais? Por quê?

l) Sinta-se a vontade para fazer comentários sobre a experiência de utilização da plataforma.

Questionário quantitativo (respondido individualmente):

m) Em relação à integração da plataforma RecS-DL ao sistema de biblioteca digital utilizado pelo seu grupo, pode-se dizer que o processo de integração é:

(5) Muito simples (4) Simples (3) Intermediário (2) Complexo (1) Muito complexo

n) Sob o ponto de vista de utilidade da plataforma RecS-DL como funcionalidade a ser incorporada a bibliotecas digitais, pode-se dizer que a plataforma é:

(5) Muito útil (4) (3) (2) (1) Pouco útil

Referências Bibliográficas

[1] Rubens Queiroz De Almeida. Software livre e inovação. *Com Ciência - Revista Eletrônica de Jornalismo Científico*, June 2004. Disponíel em http://www.comciencia.br/200406/reportagens/11.shtml. Acessado em 15 de janeiro de 2008.

[2] Amazon. Disponível em http://www.amazon.com. Acessado em 15 de janeiro de 2008.

[3] Apache Axis. Disponível em http://ws.apache.org/axis. Acessado em 15 de janeiro de 2008.

[4] Apache Tomcat. Disponível em http://tomcat.apache.org. Acessado em 15 de janeiro de 2008.

[5] Ann Apps. zetoc SOAP: A web services interface for a digital library resource. In *European Conference in Digital Libraries*, pages 198–208, 2004.

[6] David Axmark, Michael Widenius, Jeremy Cole, and Paul DuBois. *MySQL Reference Manual*. http://www.mysql.com/documentation/mysql, 2007.

[7] Andreas Bauer. PostgreSQL — Open Source Database Systems. *Linux Magazine UK*, pages 33 – 35, Feb 2002.

[8] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–44, 17 may 2001.

[9] Johan Bollen, Michael L. Nelson, Gary Geisler, and Raquel Araujo. Usage derived recommendations for a video digital library. *Journal of Network and Computer Applications*, 30(3):1059–1083, 2007.

[10] David Booth, Hugo Haas, Francis McCabe, Eric Newcomer, Michael Champion, Chris Ferris, and David Orchard. Web Services Architecture. W3C Technical Reports, published online at http://www.w3.org/TR/ws-arch/, 2004.

[11] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[12] Hung-Chen Chen and Arbee L. P. Chen. A music recommendation system based on music data grouping and user interests. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 231–238, New York, NY, USA, 2001.

[13] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. WSDL: Web services definition language. W3C Technical Reports on WSDL, published online at http://www.w3.org/TR/wsdl/, 2004.

[14] CoFE - Collaborative Filtering Engine. Disponível em http://eecs.oregonstate.edu/iis/cofe/. Acessado em 15 de janeiro de 2007.

[15] Wagner Danda da Silva Filho and Sílvio César Cazella. Star: Um framework para recomendação de artigos científicos baseado na relevância da opinião dos usuários e em filtragem colaborativa. In *ENIA 2005: Anais do Encontro Nacional de Inteligência Artificial*, pages 1042–1052, 2005.

[16] DCMI Metadata Terms. Disponível em http://dublincore.org/documents/dcmi-terms. Acessado em 15 de janeiro de 2008.

[17] Dublin Core Metadata Initiative. Disponível em http://dublincore.org. Acessado em 10 de janeiro de 2007.

[18] Erik Duval, Wayne Hodgins, Stuart A. Sutton, and Stuart Weibel. Metadata principles and practicalities. *D-Lib Magazine*, 8(4):20–27, 2002.

[19] EasyUtil Recommendation Service. Disponível em http://easyutil.com/. Acessado em 10 de janeiro de 2007.

[20] eBay. Disponível em http://www.ebay.com/. Acessado em 15 de janeiro de 2008.

[21] Keita Fujii and Tatsuya Suda. Dynamic service composition using semantic information. In *ICSOC '04: Proceedings of the 2nd international conference on Service oriented computing*, pages 39–48, New York, NY, USA, 2004.

[22] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[23] Marcos André Gonçalves. *Streams, Structures, Spaces, Scenarios, and Societies (5S): A Formal Digital Library Framework and Its Applications*. PhD thesis, Faculty of the Virginia Polytechnic Institute and State University, November 2004.

[24] Marcos André Gonçalves and Edward A. Fox. 5SL: a language for declarative specification and generation of digital libraries. In *JCDL '02: Proceedings of*

the 2nd ACM/IEEE-CS joint conference on Digital libraries, pages 263–272, New York, NY, USA, 2002.

[25] Marcos André Gonçalves, Ganesh Panchanathan, Unnikrishnan Ravindranathan, Aaron Krowne, Edward A. Fox, Filip Jagodzinski, and Lillian Cassel. The xml log standard for digital libraries: analysis, evolution, and deployment. In *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, pages 312–314, Washington, DC, USA, 2003.

[26] Nathaniel Good, Ben Schafer, Joseph Konstan, Al Borchers, Badrul Sarwar, Jonathan Herlocker, and John Riedl. Combining collaborative filtering with personal agents for better recommendations. In *American Association for Artificial Intelligence*, pages 439–446, 1999.

[27] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik Frystyk Nielsen. SOAP: Simple object access protocol. W3C Technical Reports on SOAP, published online at http://www.w3.org/TR/soap/, 2004.

[28] Erik Hatcher and Otis Gospodnetic. *Lucene in Action (In Action series)*. Manning Publications Co., Greenwich, CT, USA, 2004.

[29] Jonathan Lee Herlocker. *Understanding and improving automated collaborative filtering systems*. PhD thesis, University of Minnesota, 2000.

[30] Zan Huang, Hsinchun Chen, and Daniel Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1):116–142, 2004.

[31] Zan Huang, Wingyan Chung, Thian-Huat Ong, and Hsinchun Chen. A graph-based recommender system for digital library. In *JCDL '02: Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*, pages 65–73, New York, NY, USA, 2002.

[32] Roberto Dias Torres Júnior. Combining collaborative and content-based filtering to recommend papers. Master's thesis, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brasil, Jan 2004.

[33] Raman Kazhamiakin, Marco Pistore, and Luca Santuari. Analysis of communication models in web service compositions. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 267–276, New York, NY, USA, 2006.

[34] Choonho Kim and Juntae Kim. A recommendation algorithm using multi-level association rules. In *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on*, pages 524–527, October 2003.

[35] Mei Kobayashi and Koichi Takeda. Information retrieval on the web. *ACM Computing Surveys*, 32(2):144–173, 2000.

[36] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[37] Anísio Lacerda, Marco Cristo, Marcos André Gonçaalves, Weiguo Fan, Nivio Ziviani, and Berthier Ribeiro-Neto. Learning to advertise. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 549–556, New York, NY, USA, 2006.

[38] Carl Lagoze and Herbert Van de Sompel. The open archives initiative: building a low-barrier interoperability framework. In *JCDL '01: Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries*, pages 54–62, New York, NY, USA, 2001.

[39] Zheng Lu, Peter Hyland, Aditya K. Ghose, and Ying Guan. Using assumptions in service composition context. In *SOSE '06: Proceedings of the 2006 international workshop on Service-oriented software engineering*, pages 19–25, New York, NY, USA, 2006.

[40] Zakaria Maamar, Soraya Kouadri, and Hamdi Yahyaoui. A web services composition approach based on software agents and context. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 1619–1623, New York, NY, USA, 2004.

[41] Stuart E. Middleton, Nigel R. Shadbolt, and David C. De Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems*, 22(1):54–88, 2004.

[42] Brad Miller. The multilens cookbook. Disponível online em http://cs.umn.edu/grouplens. Acessado em 15 de janeiro de 2008., 2002.

[43] Bradley N. Miller. *Toward a Personal Recommender System.* PhD thesis, University of Minnesota, 2003. Adviser-John Riedl, University of Minesota.

[44] Bradley N. Miller, Joseph A. Konstan, and John Riedl. Pocketlens: Toward a personal recommender system. *ACM Transactions on Information Systems*, 22(3):437–476, 2004.

[45] Batul J. Mirza. Jumping connections: A graph-theoretic model for recommender systems. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, USA, Feb 2001.

[46] Batul J. Mirza, Benjamin J. Keller, and Naren Ramakrishnan. Studying recommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, 20(2):131–160, 2003.

[47] Raymond J. Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *DL '00: Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204, New York, NY, USA, 2000.

[48] NuSOAP - Web Services Toolkit for PHP. Disponível em http://dietrich.ganx4.com/nusoap/. Acessado em 15 de janeiro de 2007.

[49] Sean Owen. Taste Documentation. Disponível online em http://sourceforge.net/projects/taste/. Acessado em 15 de janeiro de 2008.

[50] Sandra Payette and Carl Lagoze. Flexible and extensible digital object and repository architecture. In *Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 21–31. Springer, 1998.

[51] Daniel Carlos Guimarães Pedronette and Ricardo da Silva Torres. RecS-DL: Uma plataforma para serviços de recomendação em bibliotecas digitais. In *XXII Simpósio Brasileiro de Banco de Dados - V Sessão de Demos*, pages 15–20, 2007.

[52] Daniel Carlos Guimarães Pedronette and Ricardo da Silva Torres. Uma plataforma de serviços de recomendação para bibliotecas digitais. In *XXII Simpósio Brasileiro de Banco de Dados - VI Workshop de Teses e Dissertações em Bancos de Dados*, pages 51–56, 2007.

[53] Saverio Perugini, Marcos André Gonçalves, and Edward A. Fox. Recommender systems research: A connection-centric survey. *Journal of Intelligent Information Systems*, 23(2):107–143, 2004.

[54] Yves Petinot, C. Lee Giles, Vivek Bhatnagar, Pradeep B. Teregowda, Hui Han, and Isaac Councill. A service-oriented architecture for digital libraries. In *ICSOC '04: Proceedings of the 2nd international conference on Service oriented computing*, pages 263–268, New York, NY, USA, 2004.

[55] Eemil Lagerspetz Petteri Nurmi, Jukka Suomela. The MobiLife Recommender. Disponível em http://www.cs.helsinki.fi/group/acs/mobilife/. Acessado em 15 de janeiro de 2008.

[56] H. Polat and Wenliang Du. Privacy-preserving top-n recommendation on horizontally partitioned data. In *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*, pages 725–731, September 2005.

[57] Andy Powell and Pete Johnston. Guidelines for implementing Dublin Core in XML. Technical Reports, published online at http://dublincore.org/documents/dc-xml-guidelines. Acessado em 15 de janeiro de 2008.

[58] Eliseo Berni Reategui and Sílvio César Cazella. Sistemas de recomendação. In *ENIA 2005: Anais do Encontro Nacional de Inteligência Artificial*, pages 306–349, 2005.

[59] Red Hat. JBoss Application Server. Disponíel em http://www.jboss.com/products/jbossas. Acessado em 15 de janeiro de 2008.

[60] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994.

[61] Paul Resnick and Hal R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, 1997.

[62] Pablo A. Roberto. Um arcabouço baseado em componentes, serviços web e arquivos abertos para criação de bibliotecas digitais. In *Anais do I Workshop em Bibliotecas Digitais*, pages 1–10, 2005.

[63] G. Salton and M. J. Mcgill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[64] Jan Schaffner and Harald Meyer. Mixed initiative use cases for semi-automated service composition: a survey. In *SOSE '06: Proceedings of the 2006 international workshop on Service-oriented software engineering*, pages 6–12, New York, NY, USA, 2006.

[65] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260, New York, NY, USA, 2002.

[66] Cyrus Shahabi and Yi-Shin Chen. An adaptive recommendation system without explicit acquisition of user relevance feedback. *Distributed and Parallel Databases*, 14(2):173–192, 2003.

[67] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating word of mouth. In *CHI '95: Proceedings of the SIGCHI confe-*

rence on Human factors in computing systems, pages 210–217, New York, NY, USA, 1995.

[68] Halvard Skogsrud, Boualem Benatallah, and Fabio Casati. A trust negotiation system for digital library web services. *International Journal Digital Libraries*, 4(3):185–207, 2004.

[69] Robert Tansley, Mick Bass, David Stuve, Margret Branschofsky, Daniel Chudnov, Greg McClellan, and MacKenzie Smith. The DSpace institutional digital repository system: Current functionality. In *JCDL '03: Proceedings of the 3th ACM/IEEE-CS joint conference on Digital libraries*, pages 87–97, 2003.

[70] Satish Thatte et al. Business Process Execution Language for Web Services, Version 1.1. Technical report, OASIS, May 2003.

[71] Roberto Torres, Sean M. McNee, Mara Abel, Joseph A. Konstan, and John Riedl. Enhancing digital libraries with Techlens+. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 228–236, New York, NY, USA, 2004.

[72] Mark van Setten, Mettina Veenstra, and Anton Nijholt. Prediction strategies: Combining prediction techniques to optimize personalization. In *Personalization in Future TV'02 at the Adaptive Hypermedia 2002 conference*, pages 78–91, Malaga, Spain, 2002.

[73] Luiz Atilio Vicentini, Regina Blanco Vicentini, and Gilmar Vicente. O acesso livre à informação científica através da biblioteca digital da unicamp: mudanças de paradigmas processo e valores na produção científica. 2006.

[74] H. Witten, Ian, Bainbridge, and David. Creating digital library collections with greenstone. *Library Hi Tech*, 23(4):541–560, January 2005.

[75] Kai Yu, Volker Tresp, and Shipeng Yu. A nonparametric hierarchical bayesian framework for information filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 353–360, New York, NY, USA, 2004.

[76] Yi Zhang. Using bayesian priors to combine classifiers for adaptive filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352, New York, NY, USA, 2004.