

# BibIme - Um Software Gerenciador de Bibliotecas Produzido de Forma Cooperativa

Dairton Bassi , Kelly Braghetto , Eduardo Colli , Fabio Kon , João Eduardo Ferreira

<sup>1</sup>Instituto de Matemática e Estatística da Universidade de São Paulo  
R. do Matão, 1010 – 05508-090 – São Paulo, SP

{dairton, kellyrb, colli, kon, jef}@ime.usp.br  
<http://malariadb.ime.usp.br/mac439/projetobiblioteca>

**Abstract.** *Many library management systems intended to manage users, books, and loans are incomplete, lack integration, or do not fully assist the needs of librarians. This paper describes the development of BibIme, a project that intends to create a complete open source solution for library management. It is based in a modularized database, where several subsystems compose the business and presentation tier. The development uses the eXtreme Programming method [Beck, 1999] and has received the help of dozens of Computer Science students at the Institute of Mathematics and Statistics of the University of São Paulo.*

**Resumo.** *Muitos sistemas gerenciadores de bibliotecas que pretendem administrar usuários, obras e empréstimos são incompletos, sem integração ou não atendem completamente às necessidades dos bibliotecários. Este artigo descreve o desenvolvimento do BibIme, um projeto que visa criar uma solução completa com código livre para o gerenciamento de bibliotecas. O BibIme é baseado em um banco de dados modularizado e vários subsistemas que compõem as camadas de negócios e apresentação. O desenvolvimento usa Programação eXtrema [Beck, 1999] como método e conta com a contribuição de dezenas de estudantes de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo.*

## 1. Introdução

Para controlar usuários, funcionários e o acervo, as bibliotecas utilizam diversos sistemas de software. Devido à grande quantidade de obras e de pessoas envolvidas no dia-a-dia de uma biblioteca, estes sistemas se tornaram fundamentais para garantir a eficiência e a boa qualidade dos serviços.

Dentro de uma biblioteca há vários setores: *Acervo, Periódicos, Gerenciamento de Usuários, Controle de Empréstimos e Controle de Aquisição*. Cada um deles requer um software que apoie suas atividades. Algumas bibliotecas já possuem seus departamentos informatizados, porém muitas vezes não há integração entre os dados, o que implica replicação de dados e impede que um sistema exiba e utilize informações dos outros, tornando suas funcionalidades limitadas. Há casos em que os sistemas trabalham de forma

interligada, mas muitas vezes a evolução de um deles é retardada ou impedida por depender de atualizações nos demais. Outro problema comum é a limitação do modelo de dados em representar algumas obras ou em comportar novos tipos. A consequência direta é o abandono ou a sub-utilização desses sistemas por serem incapazes de atender completamente às necessidades da biblioteca. O resultado indireto é a dificuldade do acesso ao que uma biblioteca tem de mais valioso: a informação.

Pensando em facilitar o controle dos empréstimos, o gerenciamento dos usuários e oferecer mais facilidades durante as consultas ao acervo, um grupo formado por professores e alunos do Instituto de Matemática e Estatística da Universidade de São Paulo (IME/USP) se uniu no segundo semestre de 2003 para trabalhar no projeto BibIme com dois objetivos: criar um banco de dados capaz de atender a todas as necessidades de uma biblioteca e produzir sistemas de software que tornem as bibliotecas mais eficientes.

Para superar as dificuldades de desenvolvimento de um software de grande porte e as complicações inerentes ao modelo de negócios das bibliotecas, dois conceitos fazem com que esse projeto contribua com a forma de se desenvolver software: a modelagem modularizada do banco de dados e o método de desenvolvimento baseado em Programação eXtrema (XP) [Beck, 1999] [Grossman et al., 2004].

O desenvolvimento ocorreu com a contribuição de dezenas de pessoas, em meio a alunos de graduação, de mestrado e professores, o desenvolvimento ocorreu em várias fases, utilizando XP durante o trabalho em equipe. Depois de um ano e meio, produzimos sistemas independentes entre si mas que interagem com uma base de dados particionada em módulos providos de funções de interface que tornam transparente toda a complexidade da modelagem.

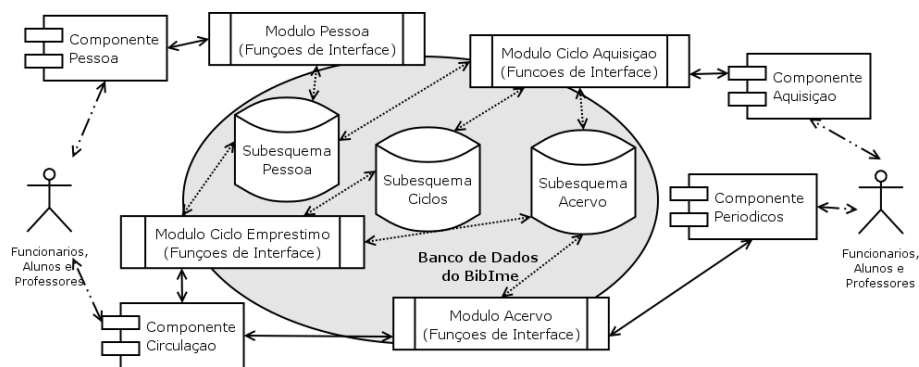
Na Seção 2 apresentamos a arquitetura do sistema. Na Seção 3 descrevemos a modelagem diferenciada do banco de dados que permite que as camadas de negócios e apresentação ofereçam diversas funcionalidades que serão detalhadas na Seção 4. Finalmente, na Seção 5 descrevemos o método usado para desenvolver o BibIme e na Seção 6 discutimos os próximos passos do desenvolvimento.

## 2. Arquitetura do Sistema

Para desenvolver sistemas que atendam a toda a gama de funcionalidades exigida por uma biblioteca, fez-se necessário criar-se uma arquitetura onde a camada de negócios e a interface gráfica são divididas em vários componentes menores que funcionam de forma independente, mas interagem com o mesmo banco de dados.

Cada partição do banco de dados chamamos de **módulo**, e, cada sistema que interage com os módulos definimos como **componente**, de forma que os componentes formam as camadas de negócios e apresentação dos módulos.

A Figura 1 mostra toda a arquitetura do BibIme. Devido ao grande escopo do projeto, neste artigo focaremos no módulo *Acervo* e no componente *Periódicos*, cuja produção está em fase avançada. O **Módulo Acervo** possui o **Componente Periódicos** constituindo suas camadas de negócios e apresentação. O componente *Periódicos* é o software usado pela equipe administrativa da biblioteca para gerenciar a parte do acervo formada por revistas científicas, revistas semanais, etc. Essas obras requerem um trata-



**Figura 1:** Arquitetura dos módulos e componentes

mento especial, pois possuem volumes e fascículos que estão associados a pagamentos freqüentes e representam uma parte da coleção que está em crescimento constante. As bibliotecas das principais universidades brasileiras investem milhões de reais por ano em revistas científicas, portanto o seu gerenciamento de forma eficiente é altamente desejável.

### 3. Banco de Dados BibIme

A modularidade de um sistema consiste na independência funcional dos seus componentes, denominados módulos [Pressman, 1992]. A consideração de módulos tornou-se um aspecto fundamental no projeto de um sistema que influi diretamente na qualidade do software final.

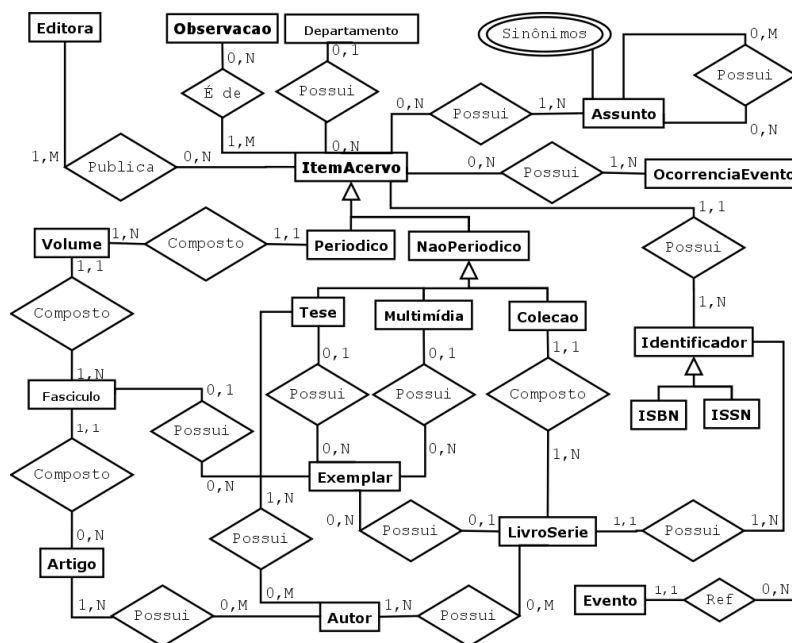
A fase de projeto de um sistema [Pressman, 1992] pode ser dividida em projeto funcional e projeto do banco de dados. A modularidade do software é tratada pelo projeto funcional, gerando vários subsistemas que acessam diretamente o banco de dados. Apesar dessa abordagem, o banco de dados continua sendo um repositório comum, armazenando os dados de forma monolítica e tirando dos subsistemas a autonomia sobre os dados.

Para obter uma autonomia parcial na administração dos dados é preciso também modularizar o banco de dados, assegurando que cada módulo seja responsável por uma parte do repositório que conterà somente os dados importantes às suas transações. A idéia básica de modularização é a decomposição do modelo global de dados da aplicação em módulos no banco de dados. O desenvolvimento modular do banco de dados do BibIme seguiu os resultados de pesquisas recentes na área [Ferreira and Busichia, 1999] [Ferreira and Finger, 2000] [Barrera et al., 2004].

#### 3.1. Modelagem

Dentro de uma biblioteca há vários tipos de obras, como livros, revistas, CDs e filmes. Os bibliotecários mantêm registros, organizam e controlam empréstimos e consultas de maneiras diferentes para cada tipo de obra. Esta forma de tratar obras parece razoável em termos logísticos, mas certamente não é a melhor abordagem para a criação de um sistema de banco de dados. Por isso nos preocupamos em diminuir os impactos da grande quantidade de tipos a fim de obter uma modelagem simplificada e generalizada.

Nossa solução se baseia na percepção de que todas as obras podem ser tratadas uniformemente, através da criação do conceito de **Item Acervo**.



**Figura 2:** Modelo conceitual do módulo Acervo, disponível em <http://malariadb.ime.usp.br/mac439/projetobiblioteca/modelos>

A estrutura associada ao *Item Acervo* é uma hierarquia de especialização. Duas especializações são *Periódicos* e *Não-Periódicos* que são separados fisicamente dentro das bibliotecas, pois possuem características próprias quanto à circulação e organização. Apesar disso, *Periódicos* e *Não-Periódicos* são subtipos de *Item Acervo*, o que permite ao banco oferecer apoio às particularidades dos itens periódicos sem comprometer ou sobrecarregar o restante do acervo.

Abaixo de *Não-Periódico*, Figura 2, há três classes de obras que representam os outros tipos de elementos encontrados na biblioteca. A flexibilidade do modelo permite que novas classes sejam facilmente acrescentadas sem afetar a estrutura atual.

Uma característica diferencial no modelo lógico do módulo *Acervo*, Figura 3, é a representação da hierarquia por um pequeno número de tabelas. O modelo torna-se flexível e pode evoluir quando novos tipos forem necessários, sem criar uma tabela para cada tipo.

### 3.2. Interface dos Módulos

A camada de interface dos módulos oferece um conjunto de *stored procedures* que tornam a modelagem transparente às camadas superiores. A complexidade fica encapsulada, facilitando o uso do banco por novos programadores. Sobretudo, as *stored procedures* garantem a flexibilidade na forma de realizar as operações sobre os dados, de modo que, se os elementos forem sempre manipulados através delas, a integridade dos dados é garantida.

Os sistemas que utilizam a base de dados fazem chamadas a funções da interface sem conhecer a estrutura física da base de dados. Essas funções encapsulam a lógica do banco de dados, permitindo que a modelagem evolua com o banco em operação, sem que seus usuários sejam afetados.

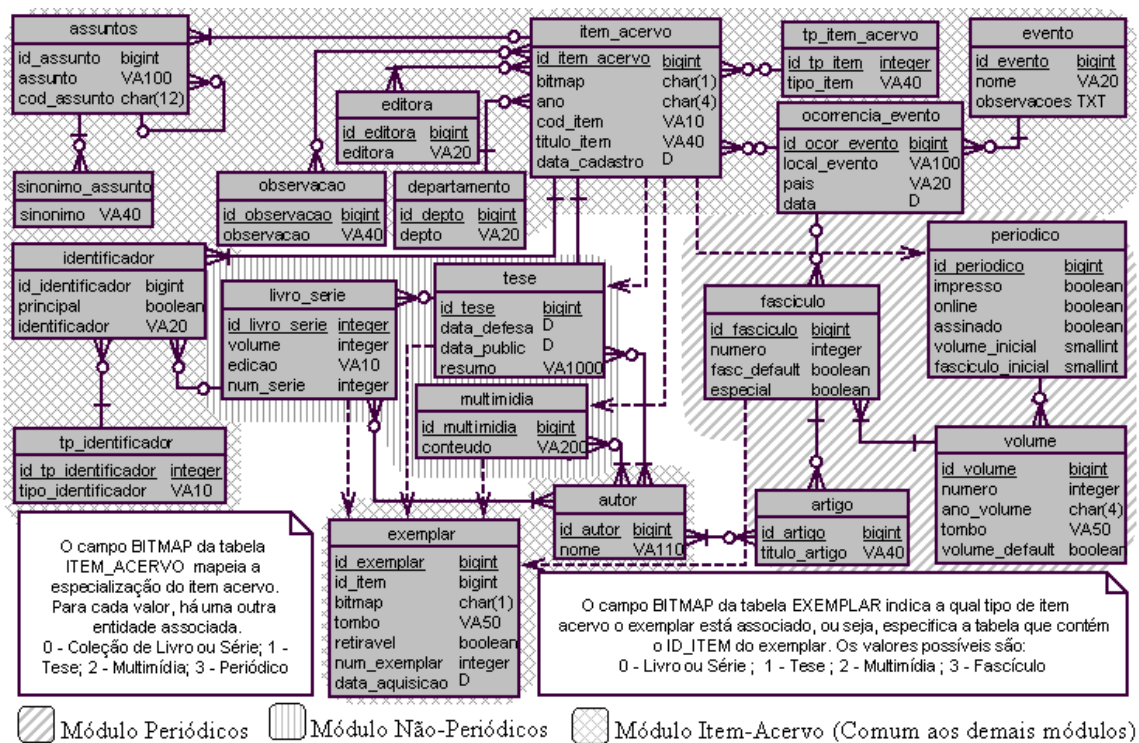


Figura 3: Modelo lógico do módulo Acervo.

#### 4. Camada de Negócios e Apresentação

Todos os componentes foram escritos em Java usando estações de trabalho portando Linux e dentro do ambiente Eclipse <sup>1</sup>, testados automaticamente com Junit <sup>2</sup> e com o CVS <sup>3</sup> como repositório e controlador de versões. Entretanto, cada módulo possui as camadas de negócios e apresentação independentes dos demais. Essa independência permite que cada um use as tecnologias que melhor acordam com suas regras de negócio.

O componente Periódicos foi construído com auxílio do Ant <sup>4</sup>, possui interface Swing seguindo o padrão Model-View-Controller (MVC) [Buschmann et al., 1996] e sua arquitetura é fortemente baseada em padrões de projeto [Gamma et al., 1995].

O componente *Periódicos* permite gravar, editar, remover e buscar por registros armazenados. Buscas mais sofisticadas podem ser feitas combinando parâmetros do periódico, dos volumes e dos fascículos e os pagamentos podem ser controlados através do sistema. Uma hierarquia de permissões dispõe operações diferentes conforme o tipo de funcionário.

Outros dois recursos se destacam devido à enorme facilidade que proporcionam aos bibliotecários: o **Relatório de Completeza** e o **Gerador de Coleção**.

Uma dificuldade muito freqüente é saber rapidamente o quão completa é a coleção de um periódico, ou seja, quais fascículos a biblioteca possui. O *Relatório de Completeza*

<sup>1</sup>http://www.eclipse.org

<sup>2</sup>http://www.junit.org

<sup>3</sup>http://www.cvshome.org

<sup>4</sup>http://ant.apache.org

calcula e apresenta de forma clara toda a coleção dos periódicos permitindo visualizar com precisão os fascículos e volumes ausentes ou incompletos.

Um processo lento e trabalhoso é fazer o cadastro de volumes e fascículos dos periódicos, pois cada periódico pode possuir dezenas de volumes e centenas de fascículos. Com o *Gerador de Coleção* os esforços durante a fase de cadastro são minimizados, o bibliotecário apenas informa a quantidade de volumes e o número de fascículos por volume para gerá-los automaticamente.

## 5. Metodologia de produção de Software Livre

A criação do BibIme é um ótimo exemplo de software cooperativo. Dezenas de pessoas contribuíram, inicialmente colhendo requisitos e modelando o banco de dados, depois desenvolvendo a interface e aplicações que persistem dados. O método consiste de um ciclo composto por duas etapas: a **Fase Incremental**, que tem duração de um semestre e ocorre a partir da segunda metade do ano, e a **Fase de Ajustes**, que começa quando a incremental acaba.

Durante os segundos semestres de 2003 e 2004 aconteceram fases incrementais, nas quais os professores adaptam partes do BibIme aos projetos de duas disciplinas de graduação do IME: Laboratório de Banco de Dados (Lab\_BD) e Laboratório de Programação eXtrema (Lab\_XP), de forma que no fim do semestre os alunos concluam uma parte do BibIme.

Um fator de dificuldade extra nesta fase é o desenvolvimento em paralelo em duas disciplinas. A primeira cria os módulos do banco de dados, a outra produz os componentes para a camada de negócios e interface desses módulos. Os desafios impostos pela concorrência das disciplinas ficaram acentuados no primeiro semestre do projeto (2003). O Lab\_BD desenvolveu os módulos *Pessoa* e *Acervo* e o Lab\_XP o componente *Periódicos* e parte do componente *Circulação*, que são partes diretamente relacionadas e exigiram muitas iterações entre as equipes para sincronizar o desenvolvimento. Em 2004, as disciplinas dedicaram-se à produção do módulo *Aquisição* e do componente *Pessoa*.

No Lab\_BD, o ponto de partida foi a coleta de requisitos e a modelagem do banco de dados, tomando como base a biblioteca Carlos Benjamin de Lyra, do IME/USP, que possui uma extensa coleção de livros e periódicos de Computação e o segundo maior acervo de Matemática da América Latina. Nesta disciplina, os alunos apresentam as modelagens diversas vezes, até que alcancem um grau de elegância e sofisticação aceitáveis em um sistema deste porte. Este processo foi auxiliado pelo uso do arcabouço *Naked Objects* [Pawson and Matthews, 2002].

No Lab\_XP, cada um dos componentes *Pessoa*, *Periódicos* e *Circulação* foi desenvolvido por uma equipe de aproximadamente oito alunos que trabalhou praticando XP [Beck, 1999]. A programação pareada, design simples, produção de testes automatizados e constantes refatorações [Fowler, 1999] no código foram algumas das práticas mais usadas e que se mostraram bastantes eficazes em manter o código com boa qualidade.

Para garantir que a produção seguisse o rumo esperado, a presença freqüente do *Cliente* [Beck, 1999] (bibliotecários e o Presidente da Comissão da Biblioteca) foi fun-

damental. Constantemente, membros da biblioteca acompanhavam o desenvolvimento para explicar as funcionalidades necessárias e esclarecer dúvidas inerentes à lógica de negócios.

Quando as disciplinas se encerram, começa a fase de *Ajustes*, pela qual o **Laboratório de Banco de Dados do IME/USP**<sup>5</sup> é responsável. No laboratório, os alunos trabalham em diferentes pontos do sistema fazendo testes, pequenas correções ou produzindo sistemas complementares para que os novos módulos e componentes entrem em produção. Desta forma, o desenvolvimento segue um ciclo que constantemente agrega código novo ao projeto. Em geral, em um semestre, novos componentes são desenvolvidos e, no semestre seguinte, na fase de *Ajustes*, estes componentes são refinados para entrar em produção na biblioteca.

Já no início de 2005, o banco de dados entrou em produção junto com o componente Periódicos. Este resultado foi possível graças à dedicação de dois alunos de mestrado - um deles desenvolvendo uma camada de software para a persistência e recuperação dos dados e o outro como administrador do banco de dados. Eles trabalham constantemente no projeto BibIme corrigindo eventuais falhas de programação, fazendo atualizações periódicas e garantindo que o sistema fique sempre em operação. Desta forma, adquiriram um vasto conhecimento sobre o projeto e tornaram-se aptos a orientar novas equipes que comecem a trabalhar no BibIme.

A adoção desta forma de desenvolvimento, baseada em disciplinas semestrais, gera dois problemas: a falta de pessoas que conheçam bem todos os detalhes do código e a falta de uma equipe fixa, o que exige maior dedicação dos novos grupos que iniciam sub-projetos, pois precisam aprender sobre o código das fases precedentes. O primeiro pôde ser abrandado mantendo uma boa documentação no código e fora dele. O segundo foi sanado com a alocação dos alunos de mestrado que já acompanham o projeto desde o início e adquiriram conhecimento amplo sobre cada módulo. Equipes temporárias foram vantajosas pois dificilmente estavam desmotivadas já que trabalharam no projeto por apenas um semestre. Como se trata de um projeto vinculado a uma disciplina, há uma forte cobrança por parte do docente para que a qualidade do software se mantenha elevada. Sobre tudo, os alunos sabem que, se não entregarem um software de alto nível, sua avaliação na disciplina estará comprometida.

## 6. Conclusões e Trabalhos Futuros

A forma com que o BibIme foi desenvolvido não exigiu nenhum novo investimento da universidade, pois usamos somente recursos já disponíveis. O grande motivo da realização deste projeto foi a união de duas necessidades: a biblioteca precisava de um software e os alunos precisavam de um projeto. Nosso trabalho se baseou em coordenar e orientar pessoas que sabiam programar para produzirem um software útil e necessário. O resultado foi uma aplicação que custaria à universidade centenas de milhares de reais.

A biblioteca ganhou um sistema que permitirá melhorar a qualidade dos serviços e os alunos tiveram a oportunidade de participar de um projeto real, onde aprenderam tanto ou mais do que se tivessem desenvolvido um software sem um uso específico. Esta

---

<sup>5</sup><http://malariadb.ime.usp.br/labd>

característica foi uma grande motivação pois todos queriam ver, pela primeira vez, o fruto do seu trabalho sendo usado por centenas de pessoas.

Motivados pelos bons resultados, pretendemos seguir com a produção de novos módulos e componentes e acrescentar funcionalidades aos que já existem. Nossos próximos passos serão concluir os componentes *Pessoa* e *Empréstimo* e, em seguida, iniciar o desenvolvimento de componentes que gerem arquivos XML do acervo no formato MARC 21 [MARC XML, ].

Acreditamos que este seja apenas o ponto inicial da produção de muitos softwares. Esperamos que outras bibliotecas usem o BibIme e assim mais cientistas e bibliotecários possam colaborar com o seu crescimento. A versão atual do código e a documentação podem ser obtidos em: <http://malariadb.ime.usp.br/mac439/projetobiblioteca>.

## Referências

- Barrera, J., Junior, R. M. C., Ferreira, J. E., and Gubitoso, M. D. (2004). An environment for knowledge discovery in biology. *Journal of Computers in Biology and Medicine*, 34(5):427–447.
- Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Addison-Wesley.
- Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., and Stad, M. (1996). *Pattern-Oriented Software Architecture — A System of Patterns*. John Wiley Press.
- Ferreira, J. E. and Busichia, G. (1999). Database modularization design for the construction of flexible information systems. *Proceedings IEEE for the IDEAS99*, pages 415–422.
- Ferreira, J. E. and Finger, M. (2000). Controle de concorrência e distribuição de dados: Teoria clássica, suas limitações e extensões modernas. *XII Escola de Computação*, pages 73–83.
- Fowler, M. (1999). *Refactoring: Improving the Design of Existing Code*. Addison-Wesley.
- Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns*. Addison-Wesley.
- Grossman, F., Bergin, J., Leip, D., Merritt, S., and Gotel, O. (2004). One xp experience: Introducing agile (xp) software development into a culture that is willing but not ready. *IBM Centre for Advanced Studies Conference*, pages 242–254.
- MARC XML. The library of congress. <http://www.loc.gov/standards/marcxml>. Último acesso em: 11/04/2005.
- Pawson, R. and Matthews, R. (2002). *Naked Objects*. Wiley and Sons.
- Pressman, R. S. (1992). *Software Engineering: a Practitioner's Approach*. McGraw-Hill, Inc.