Aplicando Padrões de Projeto na Criação de um *Framework* para uma Clínica Médica

Simone Nasser Matos[1,2], Rogério Ranthum[1], Antonio Carlos de Francisco[1], Clovis Torres Fernandes[2]

[1]Centro Federal de Educação Tecnológica do Paraná (CEFET-PR)– Unid. Ponta Grossa
Caixa Postal 20 – CEP 84016-210 – Ponta Grossa – PR

[2]Instituto Tecnológico de Aeronáutica – ITA

CEP 12228-900 – São José dos Campos – SP

{simone, rogério, acfrancisco}@pg.cefetpr.br , clovis@ita.br

Abstract. *Developing frameworks in order to create applications is not a trivial task. These frameworks generally are produced by experts because they have deep knowledge of application domain and a large experience on software design. By describing how was to apply and document the J2EE Design Patterns in the development of a framework for practices medicine, this article aims and objectives not only to highlight practical examples of design patterns applications but also to show the benefits of its use.*

Resumo. *Desenvolver frameworks para criar aplicações não é uma tarefa trivial, pois geralmente são produzidos por projetistas especializados que possuem um profundo conhecimento do domínio da aplicação e longa experiência de projeto de software. Por isso, este artigo descreve como foi aplicado e documentado os Padrões de Projeto J2EE no desenvolvimento de um framework para uma clínica médica, além disso, mostra os benefícios alcançados com seu uso.*

1. Introdução

O Prontuário Eletrônico do Paciente (PEP) é um conjunto de documentos padronizados, ordenados e concisos, destinados ao registro dos cuidados médicos prestados ao paciente em um hospital ou clínica médica (Costa 2001).

Através da criação de um PEP pode-se obter uma melhoria dos processos administrativos e financeiros, podendo-se ainda realizar uma avaliação de qualidade dos serviços prestados. Suas vantagens são: acesso remoto e simultâneo, assistência a pesquisas, diversas opções de saída de dados, diversidade nos relatórios e dados atualizados com maior freqüência (Ginneken and Moorman 1997).

De acordo com pesquisa realizada entre os anos de 2000 e 2001 pelo Núcleo de Informática Biomédica da Universidade Estadual de Campinas (NIB/UNICAMP) (Costa 2001), verificou-se que muitos sistemas na área médica não utilizam padrões de desenvolvimento. Apesar de muitas empresas possuírem equipes aptas e especializadas na confecção de *software* de qualidade, poucas têm uma visão definida do processo de desenvolvimento e acabam gerando programas de má qualidade técnica, sem atingir as expectativas esperadas dos seus usuários.

Por isso, este trabalho descreve como foi aplicada e documentada a etapa de Aplicação de Padrões de Projeto, proposta por Silva e Price (1998), apresentando exemplos práticos da utilização de Padrões de Projeto J2EE (Bond et al. 2003, Allen and Bambara 2002) na construção de regras de negócios estáveis a serem aplicadas para refinar e adequar um PEP.

Este trabalho não descreve na integra o processo de desenvolvimento do *framework*, mas sim elicita as etapas de Aplicação de Projeto J2EE, bem como apresenta as dificuldades encontradas e os benefícios alcançados, visto que em Silva e Price (1998) essas etapas não são estabelecidas.

Este artigo tem a seguinte organização. Na Seção 1, apresenta-se a motivação de se aplicar Padrões de Projeto J2EE para refinar e adequar o PEP, objetivando a criação de um *framework*. Na Seção 2, descreve-se o processo de desenvolvimento de *framework*s. Na Seção 3, apresenta-se uma breve descrição sobre Padrões de Projeto e Padrões de Projeto J2EE. As atividades para a aplicação de Padrões de Projeto J2EE na construção do *framework*, a ser utilizado por uma clínica médica, são relatadas na Seção 4. Os resultados alcançados e as dificuldades encontradas estão descritos na Seção 5. Na última Seção, descrevem-se as conclusões finais deste trabalho.

2. Processo de Desenvolvimento de *Frameworks*

Frameworks são definidos como um projeto reutilizável de uma parte ou de todo um sistema, que é representado por um conjunto de classes abstratas e concretas e pelo modo que elas interagem (Johnson 1993).

Existem várias metodologias relatadas na literatura, tais como a Dirigida por Exemplos (Johnson 1993) e a Dirigida por Pontos de Flexibilização (Pree 1995), para o desenvolvimento de *frameworks* orientado a objetos (Fayad et al. 1999). De acordo com essas metodologias, pode-se verificar que as três grandes fases do desenvolvimento de um *framework* são Análise de Domínio, Projeto e Instanciação do *Framework*.

Através da Análise de Domínio tenta-se descobrir os requisitos do contexto da aplicação alvo e os possíveis requisitos futuros, que são capturados através de experiências publicadas, sistemas de *software* similares, experiência de pessoas e padrões (Johnson 1993, Fayad et al. 1999).

A fase de Projeto do *Framework* define suas abstrações (Johnson 1993, Fayad et al. 1999). Pontos de flexibilização da estrutura (*hot spots*) e pontos de congelamento da estrutura (*frozen spots*) são modelados (Pree 1995). Para Silva (1998), esta fase passa por um conjunto de atividades, não seqüenciais, que se repetem até alcançar uma estrutura de classes que satisfaça os requisitos de generalidade, flexibilidade e extensibilidade. As etapas propostas por Silva e Price (1998) são as seguintes: Generalização, Flexibilização, Aplicação de Metapadrões, Aplicação de Padrões de Projeto e a Aplicação de Princípios Práticos de Orientação a Objetos.

Finalmente, na fase de Instanciação, os pontos de flexibilização do *framework* são implementados, gerando um sistema de *software* (Johnson 1993, Fayad et al. 1999).

3. Padrões de Projeto e Padrões J2EE

Padrão pode ser definido como o encapsulamento da descrição abstrata e estruturada de uma solução satisfatória para um problema que ocorre repetidamente dentro de um contexto, dado um conjunto de forças ou restrições que atuam sobre o problema (Gamma et al. 1994, Alexander 1977). Segundo Gamma et al. (1994), um Padrão de Projeto refere-se a problemas encontrados na etapa de projeto detalhado de *software* orientado a objetos.

Cada padrão contém vários elementos descritivos (Gamma et al. 1994, Alexander 1977). Em primeiro lugar, é definido o contexto em que ele se enquadra. Normalmente a descrição do contexto contém referências para padrões de maior escala, aplicados anteriormente, que interferem diretamente no estabelecimento do contexto do padrão. Depois há um relato do problema genérico encontrado, normalmente através de uma situação concreta para exemplificar o problema, e uma lista de restrições que são levadas em consideração para resolução do problema. Após, a solução para o problema ser apresentada, são colocadas referências a outros padrões da linguagem que o completam e que podem resolver problemas gerados ou não resolvidos pelo padrão em questão.

O termo Padrões de Projeto J2EE é usado como referência a um conjunto de padrões que têm sido identificados dentro de soluções com base na plataforma J2EE, para tratar de problemas comuns encontrados na criação de sistemas distribuídos modernos (Bond et al. 2003, Allen and Bambara 2002).

4. Aplicando Padrões de Projeto J2EE na Criação do *Framework* para uma Clínica Médica

De acordo com Gamma et al. (1994), os padrões são divididos nas três categorias seguintes: Criador (*Creational*), Estrutural (*Structural*) e Comportamental (*Behavioral*). Os Padrões de Projeto J2EE acompanham essa divisão.

Conforme descrito por Silva e Price (1998), existem várias etapas no processo de desenvolvimento de *frameworks*. Um delas é a etapa de Aplicação de Padrões de Projeto, que tem como objetivo incluir as classes de um padrão selecionado na estrutura de classes em desenvolvimento. Alternativamente, também é utilizada para fazer com que classes já existentes assumam as responsabilidades correspondentes às classes do padrão de projeto.

Silva e Price (1998) não contemplam quais atividades poderiam ser seguidas por um grupo de pessoas na fase de Aplicação dos Padrões. Por esta razão, é que neste trabalho, relata-se uma experiência prática utilizada no desenvolvimento da criação do *framework* para uma clínica médica.

O ciclo de desenvolvimento baseou-se em Programação Extrema (Fowler 2003), onde a preocupação era criar as classes de testes para depois gerar as classes de aplicação. Isto permitiu a segregação das atividades e uma maior iteração com o usuário.

A aplicação dos Padrões de Projeto J2EE, fase de Projeto, foi realizada dividindo-se o trabalho nas seguintes etapas:

1. Estudo do Padrão Arquitetural *Model-View-Control* (MVC) - Para entender o funcionamento desse padrão, analisaram-se alguns exemplos práticos do *Model 1* e *Model 2* (Geary 2002). O *Model 1* consiste de um navegador, acessando diretamente as camadas *Web* de páginas JSP, enquanto o *Model 2* permite a introdução de um *servlet* de controle entre o navegador e as páginas JSP. Durante a análise verificaram-se os prós e contras da utilização de um modelo e não do outro, conforme ilustrado em Rodrigues (2004).

2. Construção da arquitetura do *framework* - Nesta etapa aplicou-se o padrão MVC na criação arquitetural do *framework* que foi desenvolvido, elaborando-se uma arquitetura flexível dividida em camadas. Isso permitiu a segregação de atividades que deveriam ser executadas pela equipe de desenvolvimento.

   A arquitetura do *framework*, ilustrada na Figura 1, foi adaptada do *Model 2* existente, utilizando-se os conceitos de algumas tecnologias disponíveis, como por exemplo EJBs e *Struts*, entre outras. Ao aplicar o *Model 2*, conseguiu-se gerenciar múltiplos visualizadores tornando o modelo fácil de manter, testar e gerenciar. Permitiu também o desenvolvimento em paralelo, pois as camadas são independentes entre si.

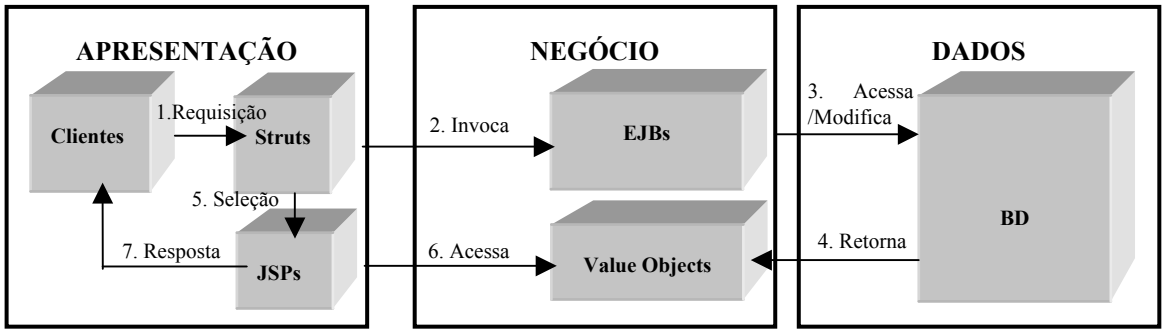

Figura 1. Modelo Detalhado do Fluxo MVC na aplicação do *framework.*

3. Identificação dos Padrões de Projeto J2EE - Neste item, pesquisaram-se os padrões estruturais que poderiam ser aplicados na estrutura de classes inicialmente desenvolvida. Foi identificada a possibilidade de aplicar os seguintes padrões: *Front Controller*, *View Helpers*, *Composite View*, *Commands*, *Dispatcher*, *Service to Worker*, *Business Delegate*, *Value Object*, *Service Locator*, *Session Façade*, *Data Access Objects* (Alur et al. 2002, Sun 2004).

4. Aplicação dos Padrões de Projeto J2EE na estrutura de classes - Neste item, realizaram-se algumas modificações nos padrões para adequá-los ao funcionamento na aplicação que estava sendo construída. Na subseção 4.1 detalha-se esta atividade.

5. Documentação de Padrões de Projeto J2EE - Para a criação da documentação do padrão escolheram-se os seguintes elementos contidos em Sun (2004): Contexto, Problema, Força, Solução, Estratégia e Conseqüências. Com o objetivo de facilitar a comunicação da equipe através de exemplos práticos da aplicação do padrão na estrutura de classes, criaram-se dois elementos, a saber: Exemplo Simples de Estrutura do Padrão na arquitetura *Model 2*; Diagrama de Seqüência

de Ações. Esses dois elementos estão baseados em informações específicas da clínica em que o *framework* será utilizado.

4.1. Aplicando Padrões de Projeto Estruturais J2EE na Estrutura de Classes

Para iniciar a aplicação dos Padrões de Projeto Estruturais, distribuiu-se cada modelo do padrão dentro das respectivas camadas, conforme ilustrado na Figura 1.

Os Padrões de Projeto Estruturais utilizados foram divididos em três categorias: Camada de Apresentação - *Front Controller*, *View Helpers*, *Composite View*, *Commands*, *Dispatcher*, *Service to Worker, Business Delegate, Value Object* e *Service Locator*; Camada de Negócio - *Session Facade*; Camada de Persistência ou Integração - *Data Access Objects*. Para cada classe criada utilizou-se um padrão de nome, como por exemplo, *FuncionarioDAO*, que representa a aplicação do padrão *Data Access Objects* para a classe *Funcionario*.

Nas subseções a seguir mostra-se um exemplo prático da aplicação dos Padrões de Projeto Estruturais J2EE em cada camada. Todas as aplicações foram experimentadas em Rodrigues (2004).

4.1.1. *Service Locator*

A classe *ExameBusinessDelegate* mapeia as regras de negócio da classe *ExameBean*. Para conseguir utilizar seus métodos, *ExameBusinessDelegate* deve fazer uma busca na rede, pois se trata de uma classe distribuída, e conceitualmente sua localização naquele exato momento não é conhecida. Toda vez que uma instância da classe *ExameBusinessDelegate*, for criada, uma busca automaticamente é feita para localizar esse objeto de negócio distribuído, conforme ilustrado na Figura 2.

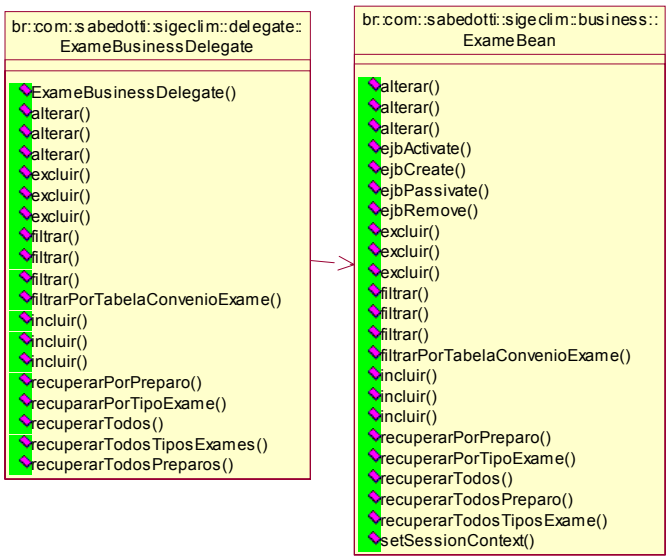

Figura 2. Exemplo do Padrão *Service Locator* na criação do *Framework*.

4.1.2. *Session Façade*

A partir do momento em que uma ação é gerada na aplicação, ela passa por uma *Business Delegate* que irá "delegar" suas atividades para a verdadeira regra de negócio da aplicação, onde as regras serão executadas e aplicadas. A execução de

uma *Session Bean* é baseada na ação de uma tecnologia de persistência de dados. No *framework*, utiliza-se o conceito de *Relation Object Model* (ROM), que é aplicado a um *framework* específico e é utilizado para fazer a persistência de dados na forma objeto-relacional, conforme explicado no diagrama de seqüência da Figura 3.

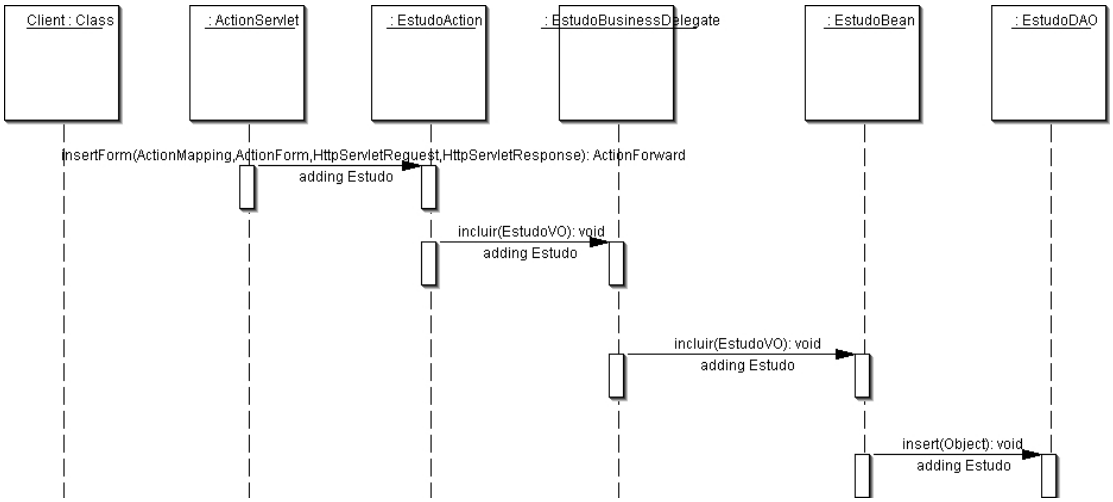

Figura 3. Exemplo de Diagrama de Seqüência do Padrão *Session Façade* na Criação do *Framework*.

4.1.3. *Data Access Object*

Como descrito na subseção anterior, o *framework* utiliza o conceito ROM para persistir seus dados. Um *framework* específico com essa característica é utilizado para mapear, via objetos XML, os atributos de uma base de dados *PostgresSQL*. Uma classe *DAO*, irá possuir implementação referente à utilização desse *framework* para persistir, recuperar e atualizar os dados provenientes de uma requisição de regra de negócio, conforme ilustrado na Figura 4.

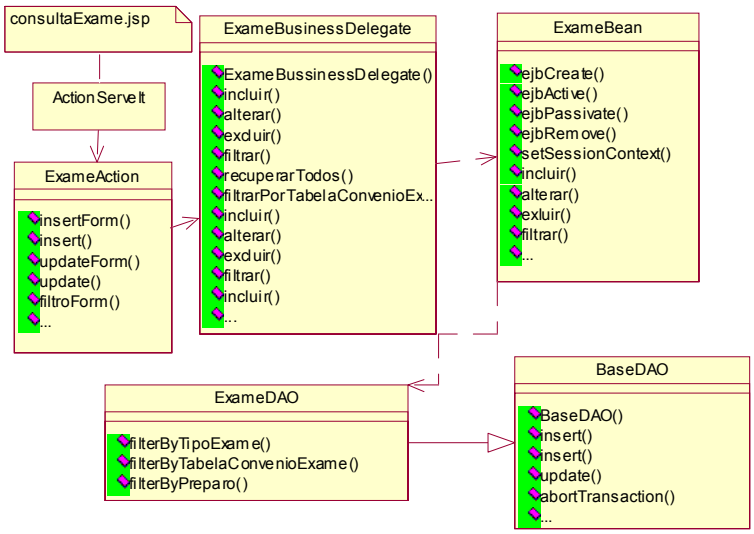

Figura 4. Exemplo do Padrão *Data Access Object* na Criação do *Framework*.

5. Resultados Obtidos

Durante a aplicação dos padrões de projeto J2EE foram identificados vários problemas como, por exemplo, dificuldade com a quantidade de exemplos práticos da utilização do padrão, pouca experiência dos desenvolvedores, necessidade de aprendizado de vários padrões e o compartilhamento de experiência entre os especialistas e novatos.

Apesar dos problemas encontrados, conseguiu-se com a aplicação dos Padrões de Projeto J2EE criar uma arquitetura mais flexível, possível de ser estendida, pois se aplicou os conceitos de MVC e camadas.

A partir do momento que foram utilizados Padrões de Projeto J2EE, conseguiram-se vários benefícios. Dentre esses benefícios, pode-se enfatizar uma maior clareza e objetividade na codificação, facilitando uma eventual refatoração na implementação, pois cada classe continha sua finalidade bem definida. Por exemplo, a classe *FuncionarioVO* possui somente os métodos *get* e *set* para cada variável de instância.

Como foram gerados documentos de contexto dos Padrões de Projeto J2EE, contendo exemplos práticos de sua aplicação na criação do *framework* específico para uma clínica médica, novas pessoas foram integradas ou substituídas no projeto sem que isso comprometesse o desenvolvimento do mesmo.

Com a utilização de objetos distribuídos, e de um servidor de aplicação, ficou fácil gerenciar a instanciação de objetos e suas interfaces assim como um *pool* de conexões para um meio de persistência de dados. Com a aplicação do conceito de Inversão de Controle, o *framework* delega várias funcionalidades ao servidor de aplicação, tornando a vida do desenvolvedor mais centrada nas regras de negócio que precisa alcançar.

É importante ressaltar que com a criação da documentação sobre padrões de projeto, houve a padronização na comunicação da equipe, segregação das atividades e de ganho significativo de produtividade. A padronização na comunicação ocorreu, pois os desenvolvedores tiveram conhecimento de exemplos práticos dos Padrões de Projeto J2EE que estavam sendo aplicados no projeto. Também adotou-se a padronização nomeando os componentes e seus *packages* de acordo com o padrão utilizado e camada específica, permitindo desta forma um melhor gerenciamento.

A segregação das atividades foi atingida depois que a equipe de desenvolvimento teve maior controle sobre a padronização de criação do sistema, dividindo-se as funcionalidades de maneira que o trabalho fosse feito independente. O ganho de produtividade foi alcançado através da segregação do trabalho e da padronização de desenvolvimento da equipe, pois cada integrante era responsável pelo desenvolvimento de uma camada específica da aplicação. Enquanto um elaborava as classes de teste, outro estava confeccionando as interfaces gráficas.

6. Conclusões

Este trabalho mostrou de uma forma prática como os Padrões de Projeto foram aplicados e documentados, visando formalizar o conhecimento, de forma a registrar de maneira estruturada as soluções que podem ser utilizadas na etapa de Aplicação de Padrões de Projeto, durante a criação de *frameworks*.

Através das atividades estabelecidas para o processo de aplicação de Padrões de Projeto, notou-se que a rotatividade de pessoas no projeto pode ocorrer sem que com isso houvesse a perda de qualidade no sistema. Pois, foram produzidas por um processo que permite a padronização de comunicação e a segregação de atividades, reduções de quantidade de código geradas, fatores estes que contribuíram para se obter um aumento de produtividade.

7. Referências

Alexander, C. (1977). "A Pattern Language: Towns, Buildings, Constructions", New York: Oxford University Press.

Allen, P., Bambara, J. (2002). Guia Oficial de Certificação J2EE. SP, Campus, p. 613.

Alur, D., Crupi, J. Dan, M. (2002). Core J2EE Patterns, Rio de Janeiro, p. 406.

Bond, M., Haywood, D., Law, D., Longshaw A.,Roxburgh, P. (2003). Aprenda J2EE com EJB, JSP, Servelts, JNDI, JDBC e XML em 21 dias, S. Paulo, M. Books, p. 962.

Costa, C.G. A. (2001) "Desenvolvimento e Avaliação Tecnológica de um Sistema de Prontuário Eletrônico do Paciente, baseado no Paradigma da World Wide Web e da Engenharia de Software", Campinas, Universidade Estadual de Campinas, Dissertação de Mestrado, p. 268.

Fayad, M., Schmidt, D., Johnson, R. (1999). Building Application Frameworks – Object-Oriented Foundations of Framework Design, Wiley & Sons, 1-100.

Fowler, Martin. (2003). "The new methodology", Disponível em: http://www.martinfowler.com/articles/newMethodology.html, Acesso em Jan/2005.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1994). "Design Patterns: Elements of Reusable Object-Oriented Software", Reading, MA: Addison-Wesley.

Geary, D. M. (2002). Java Server Pages Avançado: Plataforma Java 2 Enterprise Edition, Ciência Moderna, Capítulo 5-6, p. 430.

Ginneken, A. M., Moorman, P. W. (1997). "The Pacient Record". In: Van Bemmel, J.H., MA (eds). "HandBook of Medical Informatics", Houten, the Netherlands: Bohn Stafleu Van Loghm.

Johnson, R. E. (1993). "How to design frameworks". In: Object-Oriented Programming Systems, Languages and Aplications Conference, Washington Proceedings, 1-26.

Pree, W. (1995). Design Patterns for Object-Oriented Software Development, Addison-Wesley, Reading, Mass, p. 268.

Rodrigues, F. P. (2004). "Projeto Sigeclim – Aplicando Padrões de Projeto", Coordenação de Informática, CEFET-PR Unidade de Ponta Grossa, TD.

Silva, R. P., Price, R. T. (1998). "A busca de generalidade, flexibilidade e extensibilidade no processo de desenvolvimento de *frameworks* orientados a objetos". In: Proceedings of Workshop Iberoamericano de Engenharia de Requisitos e Ambientes de Software (IDEAS'98). Torres: apr., v.2, p.298-309.

Sun. (2004). Disponível em: http://java.sun.com/blueprints/corej2eepatterns/ . Acessado em Agosto de 2004.