Carga Dinâmica de Componentes via Biblioteca Brechó

Paula Fernandes, João Gustavo Prudêncio, Anderson Marinho, Marco Lopes, Leonardo Murta, Cláudia Werner

PESC/COPPE – Universidade Federal do Rio de Janeiro
Caixa Postal 68.511 – 21945-970 – Rio de Janeiro – RJ – Brasil

{paulacibele,gustavo,mlopes,murta,werner}@cos.ufrj.br,
andymarinho@ufrj.br

Resumo. *Este artigo apresenta uma ferramenta para carga dinâmica de componentes no ambiente Odyssey, utilizando a biblioteca de componentes Brechó. Ela foi implementada como forma de evoluir o mecanismo de carga dinâmica existente nesse ambiente. A Brechó foi utilizada para armazenar os componentes a serem instalados, possibilitando o desenvolvimento de uma solução mais flexível e organizada. O principal objetivo da ferramenta proposta é permitir que novas funcionalidades disponibilizadas através de componentes, armazenados na Brechó, possam ser adicionadas e removidas do ambiente Odyssey em tempo de execução, de acordo com a demanda dos usuários.*

1. Introdução

O ambiente Odyssey [8] tem como principal objetivo apoiar a reutilização de software por meio de técnicas de engenharia de domínio, linha de produtos e desenvolvimento baseado em componentes.

Durante a evolução deste ambiente, várias ferramentas de apoio às atividades de reutilização foram desenvolvidas. Porém, por serem fortemente acopladas ao Odyssey, afetavam negativamente características do ambiente como usabilidade, desempenho e evolução. Com a finalidade de contornar esse problema, foi realizada uma reengenharia do ambiente Odyssey, separando em um núcleo, denominado Odyssey Light, as funcionalidades identificadas como essenciais para a modelagem baseada em reutilização. As demais funcionalidades foram encapsuladas em ferramentas de apoio, denominadas *plug-ins*. Contudo, esta reengenharia demandou o desenvolvimento de um mecanismo que possibilitasse a carga dinâmica, por demanda, das ferramentas no núcleo [6], permitindo o gerenciamento de variabilidades em tempo de execução, por meio de seleção, recuperação e instalação de *plug-ins*.

No entanto, esse mecanismo de carga dinâmica possui algumas limitações. As ferramentas, suas dependências e sua localização são descritas em um arquivo XML. Para cada nova *release* do ambiente Odyssey, um novo arquivo deve ser criado apenas com as ferramentas compatíveis. Além disso, nenhuma ferramenta de apoio à edição desses arquivos foi desenvolvida, como forma de garantir sua consistência, por exemplo, em relação às ferramentas e suas dependências. Todos os arquivos binários correspondentes aos *plug-ins* e suas dependências são armazenados em um mesmo diretório em rede, não havendo nenhuma organização, dificultando sua manutenção. Outra limitação desta abordagem é o fato de determinado desenvolvedor não ter

informação sobre a utilização da sua ferramenta, uma vez que não há nenhum registro da instalação de *plug-ins* no Odyssey.

O objetivo deste artigo é apresentar uma evolução do mecanismo de carga dinâmica do ambiente Odyssey, que busca contornar os problemas citados anteriormente por meio da integração via Web Services deste ambiente com a biblioteca de componentes Brechó [1]. A Brechó é um sistema de informação para a Web que fornece mecanismos de documentação, armazenamento, busca e recuperação de componentes. Essa integração visa possibilitar que a Brechó faça a mediação entre o Odyssey e as suas ferramentas, carregadas dinamicamente.

Este artigo está organizado em cinco seções. Na Seção 2, são apresentados os principais conceitos envolvidos neste trabalho. Na Seção 3, é discutido o mecanismo proposto para variabilidade em tempo de execução por meio da carga dinâmica de componentes via Brechó. Na Seção 4, é apresentado um exemplo de utilização desse mecanismo. Por fim, na Seção 5, são destacadas as contribuições e trabalhos futuros.

2. Contextualização

Nesta seção são discutidos alguns conceitos importantes para a contextualização deste trabalho. Na Seção 2.1 é abordado o conceito de variabilidade de sistema, que é uma das principais motivações para este trabalho. Na Seção 2.2 é apresentado o mecanismo de carga dinâmica anteriormente utilizado no ambiente Odyssey, que serviu como base para a abordagem proposta. Finalmente, a Seção 2.3 apresenta brevemente a Brechó.

2.1. Variabilidade de sistema

Variabilidade de sistema é a habilidade do software ser eficientemente estendido, modificado, adaptado ou configurado para uso em um contexto particular [9]. Devido à relação custo-benefício, sistemas precisam ser liberados em diferentes distribuições (e.g. *standard* ou *professional*) [5]. Além disso, é importante permitir a configuração do sistema para se adequar às necessidades particulares dos clientes.

A variabilidade de sistema pode ocorrer em diferentes fases do ciclo de vida do software. Por exemplo, na fase de especificação, diferentes tecnologias permitem a representação da variabilidade do sistema, tais como o uso de elementos opcionais e variáveis em linhas de produtos para dar suporte à seleção de produtos [2] e à seleção de características em um modelo de domínio dentro de um processo de reutilização [7]. A variabilidade em tempo de implantação é uma das mais conhecidas, porém é menos flexível que a variabilidade em tempo de execução, visto que não é possível instalar novas funcionalidades durante a execução do software.

Além dessas abordagens voltadas para fases particulares do ciclo de vida do software, Hoek [4] e Gurp [3] sugerem o uso de arquiteturas de software para guiar a seleção de variabilidades a qualquer momento do ciclo de vida. Dessa forma, é possível definir variabilidade em tempo de projeto e aplicá-la no tempo de projeto, invocação ou execução. Em nosso caso, é importante permitir que a equipe de desenvolvimento do Odyssey descreva as variabilidades em tempo de desenvolvimento e que os engenheiros de software (usuários do Odyssey) selecionem as funcionalidades em tempo de execução. Alguns ambientes de desenvolvimento integrados, como Eclipse e NetBeans, utilizam uma abordagem baseada em *plug-ins* para atender a esse requisito.

2.2. Mecanismo anterior de carga dinâmica do Odyssey

Como mencionado na Seção 1, todas as ferramentas foram retiradas do núcleo do ambiente Odyssey e transformadas em *plug-ins*, tornando necessário o desenvolvimento de um mecanismo que possibilitasse a carga por demanda das ferramentas no núcleo.

Inicialmente, foi necessário o desenvolvimento de uma interface comum para todas as ferramentas, denominada *Tool*, que é acessada pelo ambiente Odyssey sempre que ele precisa consultar ou notificar os plug-ins. Essa interface define métodos que fornecem informações como, por exemplo, sobre quais menus devem ser incluídos no ambiente.

Nessa abordagem, uma ferramenta se torna um componente, ou *plug-in*, quando implementa a interface *Tool* e é empacotada em um arquivo JAR (*Java Archive*), armazenado em um diretório específico na Web. Todas as ferramentas, empacotadas como componentes, são descritas em um arquivo XML, contendo nome, tipo, descrição, localização no repositório e dependências.

Os tipos de componentes disponíveis são *kernel*, *plug-in* e *library*. Os componentes do tipo *kernel* estão no núcleo do ambiente Odyssey (por exemplo, o editor de diagramas). Os *plug-ins* podem ser ativados por meio de seleção, provendo variabilidade para o ambiente (por exemplo, adicionando a funcionalidade de geração de código por meio da ferramenta Odyssey-MDA-codegen). Finalmente, o tipo *library* engloba os componentes requeridos por outros componentes, mas que não representam ferramentas para o Odyssey (por exemplo, uma biblioteca específica de JDBC). Na Figura 1 temos a descrição da ferramenta Odyssey-XMI, utilizada para importação e exportação de modelos UML no formato XMI. Ela possui dependência para outros dois componentes do tipo *library*, que devem ser recuperados e instalados previamente, o que é feito de forma automática pelo mecanismo de carga dinâmica.

```
<component type="plugin" name="Odyssey-XMI-0.2.2.jar" description="Odyssey-XMI"
location="http://reuse.cos.ufrj.br/releases/components">
        <dependency name="MofRepository-1.0.0.jar"/>
        <dependency name="jmi-uml-1.4.jar"/>
</component>
```

Figura 1. Descrição da ferramenta Odyssey-XMI.

O mecanismo de carga dinâmica propriamente dito é responsável pela comunicação entre a instância do ambiente Odyssey e o diretório de componentes, incluindo a recuperação do arquivo descritor dos componentes e suas implementações, que ocorre através da utilização do protocolo HTTP. Após a fase de download, um novo *class loader* é criado para acessar as classes recuperadas. Então, a API de reflexão do Java é utilizada para acessar a classe declarada no arquivo *manifest* do componente. Essa classe é convertida para a interface *Tool* e colocada em uma coleção que contém todas as ferramentas instaladas.

2.3. A biblioteca Brechó

A biblioteca Brechó é um sistema de informação para Web que possui como principal objetivo fornecer mecanismos de documentação, armazenamento, busca, e recuperação de componentes [1].

Os mecanismos de publicação e documentação adotam um conceito flexível de componente, que vai além do arquivo binário, visando uma representação que inclua os possíveis artefatos produzidos durante o desenvolvimento do componente (como especificações, código fonte e manuais). A estrutura de documentação é fundamentada em categorias e formulários dinâmicos e configuráveis. Assim, é possível classificar um componente em várias categorias, além de criar e associar diferentes formulários de documentação a cada categoria, permitindo a construção da documentação do componente como um mosaico.

A organização interna da Brechó é dividida em níveis, que levam em consideração diferentes aspectos (cortes funcionais, temporais, etc.), para melhor representação do componente. O primeiro nível, denominado componente, representa conceitualmente as entidades armazenadas na Brechó, sem as informações concretas sobre as implementações dessas entidades. O nível distribuição representa um corte funcional sobre as entidades, fornecendo conjuntos de funcionalidades que são desejadas por grupos específicos de usuários. O nível *release* representa um corte temporal sobre as distribuições, no qual define versões dos artefatos que implementam as entidades em um determinado instante no tempo. A partir desse nível, as entidades passam a ter informações concretas sobre suas implementações, que usualmente existem em diferentes níveis de abstração (e.g. documentação do usuário, análise, projeto, código, binário). O nível pacote permite que seja feito um corte em níveis de abstração, possibilitando que sejam agrupados artefatos de acordo com o público alvo de reutilização. O nível licença possibilita a definição de níveis de serviço sobre os pacotes. Para cada pacote podem ser estabelecidas licenças específicas, que garantem regras entre os produtores e os consumidores dos componentes.

A Brechó oferece ainda outros mecanismos importantes, como, por exemplo, a representação da relação de dependência entre componentes, que é fundamental durante a recuperação de um componente, apresentando os componentes dos quais ele necessita para funcionar. Um outro mecanismo oferecido é o mapa de reutilização, que serve para auxiliar na atividade de identificação de responsabilidades de manutenção, mantendo as informações dos contratos firmados entre os produtores e consumidores, para cada componente reutilizado.

3. Carga dinâmica de componentes via Brechó

Conforme apresentado na Seção 1, o mecanismo de carga dinâmica anteriormente presente no ambiente Odyssey possui algumas limitações. Grande parte desse mecanismo tem como base o arquivo XML com a descrição das ferramentas e suas dependências. Esse arquivo é gerado de forma estática para cada nova *release* do ambiente Odyssey sem nenhuma ferramenta de apoio, que ajude a garantir sua consistência. Todos os arquivos para instalação dos componentes são armazenados de forma aleatória em um diretório em rede, dificultando sua manutenção. Além disso, os desenvolvedores não possuem nenhum controle da utilização da sua ferramenta.

Como forma de superar essas limitações, o mecanismo proposto neste artigo para carga dinâmica de componentes no ambiente Odyssey utiliza algumas funcionalidades da Brechó. Para a integração Odyssey-Brechó optou-se pelo uso da tecnologia Web Services.

Foram criados na Brechó os serviços *UserService* e *ComponentService*, ilustrados na Figura 2, que disponibilizam parte das funcionalidades da biblioteca para uso por outras ferramentas. O serviço *UserService* possui métodos relacionados à autenticação dos usuários na Brechó, tornando possível a identificação dos consumidores dos componentes cadastrados. O serviço *ComponentService* engloba métodos relacionados à manipulação dos componentes, desde a obtenção da sua descrição até a sua recuperação, que são detalhados a seguir.

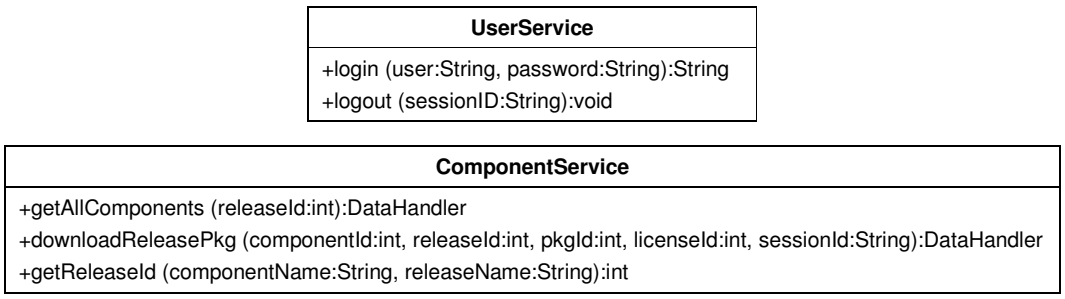

Figura 2. Serviços *UserService* e *ComponentService*.

O arquivo descritor dos componentes é gerado dinamicamente a partir da chamada ao método remoto *getAllComponents()*. Esse método possui como parâmetro o identificador da *release* do componente Odyssey, obtido por meio da chamada ao método remoto *getReleaseId(),* através da qual o mecanismo realizou a chamada remota. Ele retorna um arquivo XML com a descrição dos componentes e suas respectivas *releases* que dependem da *release* do componente Odyssey especificada. Além disso, esse arquivo contém a descrição das dependências diretas e indiretas desses componentes.

Devido à organização interna dos componentes na Brechó, foi necessário realizar algumas modificações em relação à forma como os componentes eram descritos no mecanismo anterior. A Figura 3 ilustra a nova descrição da ferramenta Odyssey-XMI.

```
<component id="2" name="Odyssey-XMI" description="Ferramenta Odyssey-XMI"
category="plugin">
   <release id="3" name="0.2.2">
      <package id="2" name="Default">
         <license id="1" name="Default" />
      </package>
      <dependency compid="3" relid="5" relname="1.0.0" pkgid="4" licid="1" />
      <dependency compid="4" relid="4" relname="1.4" pkgid="3" licid="1" />
   </release>
</component>
```

Figura 3. Nova descrição da ferramenta Odyssey-XMI.

O método *downloadReleasePkg()* possui como parâmetros os identificadores de componente, *release*, pacote e licença do pacote que deve ser recuperado, além do identificador do usuário para registro no mapa de reutilização.

Para que os componentes e suas dependências possam ser acessados e posteriormente instalados no ambiente Odyssey, eles devem ser previamente cadastrados na Brechó sob algumas diretrizes: (i) eles devem ser cadastrados nas categorias que identificam seu tipo (*kernel*, *library* ou *plugin*); (ii) suas *releases* devem possuir um artefato contendo o arquivo JAR que será utilizado no processo de instalação

no ambiente Odyssey e um pacote que contenha esse artefato. Além disso, é necessário estabelecer a relação de dependência entre as *releases* dos componentes do tipo *plug-in* com as *releases* do componente Odyssey cadastradas na Brechó com as quais são compatíveis. Essa é uma informação fundamental para o funcionamento do mecanismo proposto, pois ela é utilizada para identificar os componentes disponíveis para instalação em uma determinada *release* do componente Odyssey.

4. Exemplo de utilização

A Figura 4 mostra o processo de disponibilização da *release* de um componente na Brechó. A Brechó permite a utilização de níveis do tipo padrão (*default*) para distribuição, pacote e licença, que facilitam esse processo.

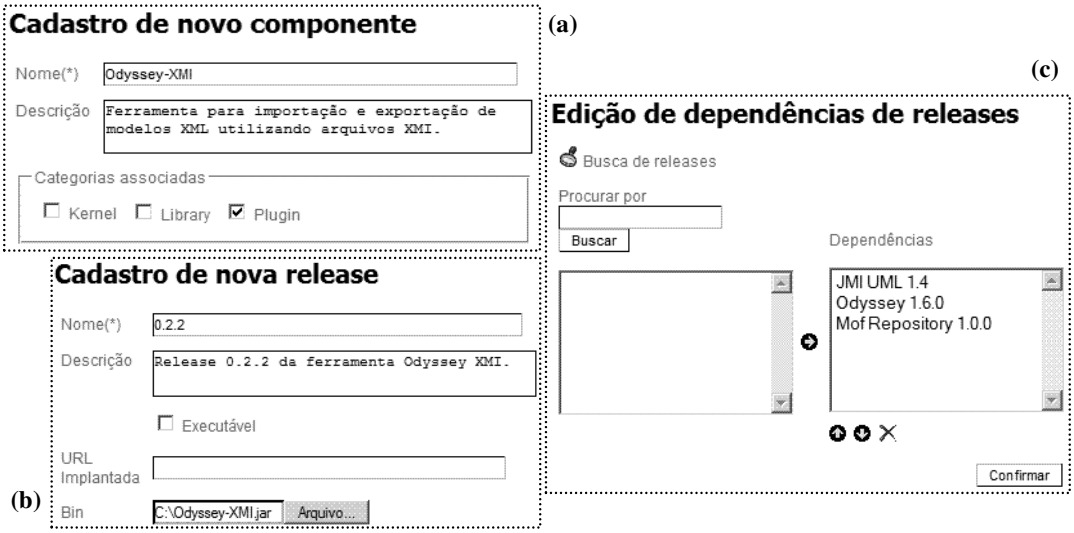

Figura 4. Cadastro de um componente na biblioteca Brechó.

Inicialmente, é realizado o cadastro do componente e a escolha da sua categoria (Figura 4.a). Depois é realizado o cadastro da *release*, que passa a ter as informações concretas em relação à implementação do componente (Figura 4.b). Nessa etapa, é adicionado o arquivo JAR do componente. Por fim, é realizada a definição das dependências da *release* cadastrada (Figura 4.c).

No ambiente de configuração do Odyssey, o usuário pode definir a URL da biblioteca Brechó em que ele deseja acessar os componentes, o diretório local no qual esses componentes devem ser armazenados após a recuperação, além do usuário e senha utilizados para autenticação na Brechó (Figura 5.a). Após preencher esses dados, o usuário pode listar os componentes disponíveis e escolher se quer instalá-los ou desinstalá-los (Figura 5.b). Nessa etapa, algumas informações sobre os componentes podem ser visualizadas como, por exemplo, o tipo, o status e os componentes dos quais ele depende diretamente. No nosso exemplo, temos as informações da ferramenta Odyssey-XMI, que depende diretamente de outros dois componentes. Depois que a opção de instalação ou desinstalação é selecionada, o *wizard* de integração informa os componentes que serão instalados ou desinstalados, incluindo suas dependências diretas e indiretas (Figura 5.c). A ferramenta Odyssey-XMI possui dependência indireta para outros seis componentes, ou seja, para que suas dependências diretas possam ser

instaladas, outros componentes devem ser instalados concomitantemente. Finalmente, no caso da instalação, o *wizard* recupera os componentes e os carrega dinamicamente no ambiente, tornando a ferramenta acessível através do *menu* Tools. No caso da instalação da ferramenta Odyssey-XMI, o ambiente Odyssey passa a fornecer a funcionalidade de importação e exportação de modelos UML no formato XMI.

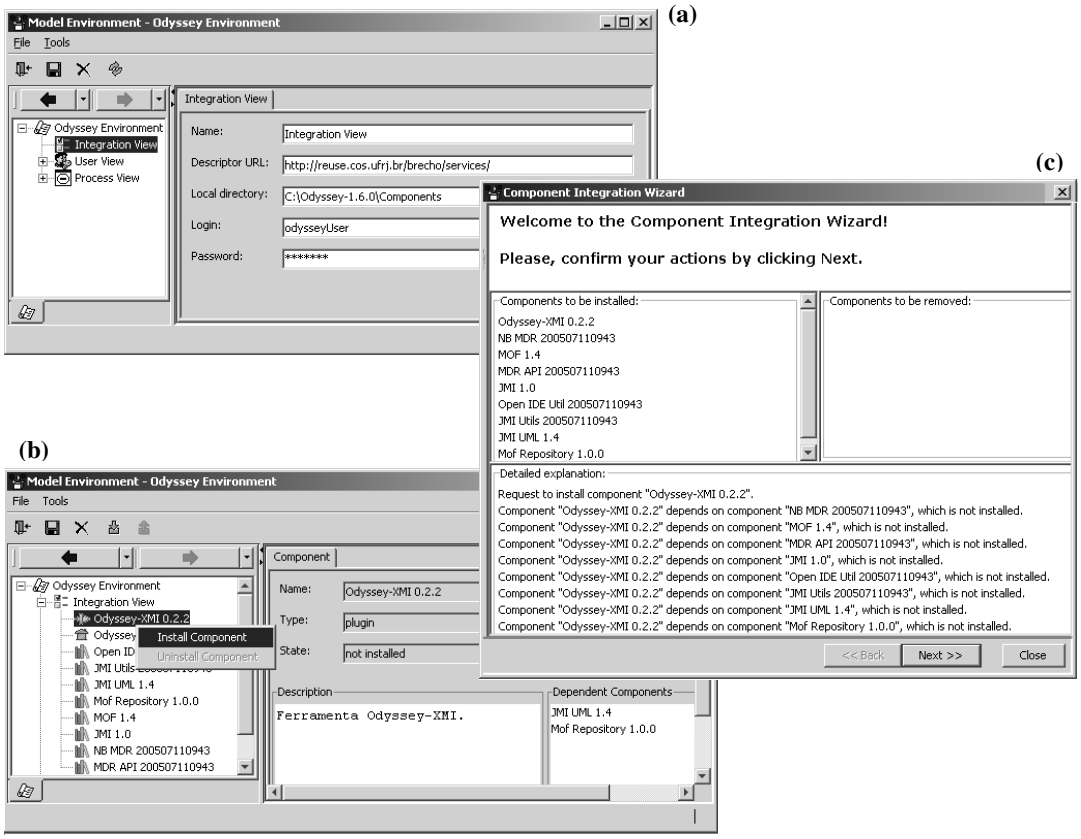

Figura 5. Carga dinâmica de um componente no ambiente Odyssey via Brechó.

5. Conclusões

Esse artigo apresentou uma ferramenta para carga dinâmica de componentes dentro de um ambiente em execução, utilizando uma biblioteca de componentes (i.e., Brechó). Essas ferramenta foi projetada e implementada para prover uma solução mais eficiente e flexível para o mecanismo de carga dinâmica do Odyssey [6]. Ela foi desenvolvida sem a necessidade de modificar a estrutura interna das ferramentas, utilizando como base parte dos mecanismos existentes. Além disso, permite um maior controle sobre a evolução dos componentes que podem ser instalados no Odyssey, facilitando o processo de geração de novas *releases* desse ambiente.

A carga dinâmica de componentes via Brechó trouxe benefícios tanto para os usuários do Odyssey quanto para os produtores das ferramentas. Para os primeiros, é possível acessar componentes sempre atualizados para serem utilizados de acordo com suas demandas em tempo de execução. Já para os produtores, o processo de disponibilização das suas ferramentas tornou-se mais simples e organizado, além de

permitir que o produtor tenha conhecimento sobre os seus consumidores, podendo notificá-los sobre, por exemplo, a liberação de novas versões.

Como trabalhos futuros, vislumbramos o tratamento da relação de conflito entre componentes, possibilitando a identificação de componentes incompatíveis e evitando o mau funcionamento dos componentes. Além disso, existe a possibilidade de tornar esse mecanismo de carga dinâmica sensível ao contexto, de forma que componentes possam ser automaticamente instalados sem a intervenção do usuário, com base nas ações do usuário dentro do ambiente Odyssey.

A Brechó pode ser acessada no link *http://reuse.cos.ufrj.br/brecho*. O ambiente Odyssey, na *release* 1.6.0, compatível com o novo mecanismo de carga dinâmica, pode ser obtido na própria Brechó.

Agradecimentos

Os autores gostariam de agradecer à CAPES e ao CNPq pelo apoio financeiro, e à equipe de Reutilização da COPPE/UFRJ, que direta ou indiretamente apoiaram na realização desse trabalho.

Referências

[1] BRECHÓ (2007) "Projeto Brechó". In: http://reuse.cos.ufrj.br/brecho, acessado em 08/06/2007.

[2] GARG, A., CRITCHLOW, M., CHEN, P., *et al*. (2003) "An Environment for Managing Evolving Product Line Architectures". In: *International Conference on Software Maintenance*, pp. 358-367, Amsterdam, Netherlands, September.

[3] GURP, J., BOSCH, J., SVAHNBERG, M. (2001) "On the Notion of Variability in Software Product Lines". In: *Proceedings of the Working IEEE/IFIP Conference on Software Architecture*, pp. 45-54, Amsterdam, Netherlands, August.

[4] HOEK, A. (2004) "Design-Time Product Line Architectures for Any-Time Variability", *Science of Computer Programming*, v. 53, n. 3 (December), pp. 285-304.

[5] LEON, A. (2000) *A Guide to Software Configuration Management*, Artech House Publishers.

[6] MURTA, L., VASCONCELOS, A., BLOIS, A., *et al*. (2004) "Run-time Variability through Component Dynamic Loading". In: *XVIII Simpósio Brasileiro de Engenharia de Software, Sessão de Ferramentas*, pp. 67-72, Brasília, DF, Brasil, Outubro.

[7] MILER, N., WERNER, C., BRAGA, R. (2000) "O uso de Modelos de Features na Engenharia de Aplicações". In: *IDEAS´00*, pp. 85-96, Cancun, México, April.

[8] ODYSSEY (2007) "Projeto Odyssey". In: http://reuse.cos.ufrj.br/odyssey, acessado em 08/06/2007.

[9] SVAHNBERG, M., GURP, J., BOSCH, J. (2005) "A taxonomy of variability realization techniques", *Software: Practice and Experience*, v. 35, n. 8 (April), pp. 705-754.