DiretoJ2EE: uma Modelagem J2EE para o Direto

Karina K. Silveira[1] **, Sérgio L. B. Vidal**[1] **, Flávio R. Maciel**[1] **, Rômulo B. Rosinha**[1] **, Cláudio F. R. Geyer**[1]

[1] Instituto de Informática – Universidade Federal do Rio Grande do Sul
Av. Bento Gonçalves, 9500 - Campus do Vale - Bloco IV
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

{kohl,vidal,frm,romulo,geyer}@inf.ufrgs.br

Abstract. *This article describes the studies about a J2EE implementation for Direto, the free, open source enterprise mail, scheduling and directory software developed by PROCERGS. The goal of these studies is to achieve better modular, reusable and scalable software.*

Resumo. *Este artigo descreve os estudos de implementação da arquitetura do Direto, software livre de correio, agenda e catálogo corporativos desenvolvido pela PROCERGS, para a especificação Java 2, Enterprise Edition (J2EE). Estes estudos foram desenvolvidos visando obter melhor modularidade, reusabilidade e escalabilidade ao software em questão.*

1. O Direto

O Direto [DiretoGNU, 2003] é um *software* de agenda, catálogo e correio corporativo desenvolvido pela PROCERGS, Companhia de Processamento de Dados do Estado do Rio Grande do Sul, com o objetivo de atender os diversos órgãos do estado. A PROCERGS optou pelo desenvolvimento interno de uma ferramenta de *groupware*, pois dessa forma seria permitida a evolução gradual e atendimento das necessidades especificas dos usuários. Além disso, o desenvolvimento de uma solução própria é de grande atrativo em relação a custos e flexibilidade da solução. Esse baixo custo justifica-se pelo Direto possuir licença de *software* livre e basear-se totalmente em tecnologias baseadas na filosofia de *software* livre, tais como Java, Apache, Tomcat, OpenLDAP e PostgreSQL.

Porém, como constatado em estudos já realizados [Geyer et al., 2001], o sistema peca na questão da modularidade, por não usar da melhor forma o paradigma orientado a objeto do qual tem poder com o uso da linguagem Java. Os componentes de *software* do Direto possuem um alto grau de acoplamento entre si, o que torna a sua manutenção difícil e o prejudica em relação à escalabilidade [Maciel, 2002].

A partir da proposta de portar o Direto para o ambiente J2EE [Sun Microsystems, 2003a], alguns trabalhos foram realizados, como a modelagem e prototipação da Agenda [Rosinha, 2002] [Silveira, 2003], Catálogo [Maciel, 2002] e Correio [Vidal, 2003]. Este trabalho apresenta alguns detalhes sobre as tecnologias utilizadas e modelagens realizadas.

2. As Tecnologias

A plataforma J2EE é um conjunto de tecnologias de suporte ao desenvolvimento de aplicações corporativas distribuídas. Ela implementa serviços de baixo nível que tendem a facilitar a reutilização

de componentes em diferentes aplicações e que permitem ao desenvolvedor concentrar-se principalmente na implementação da lógica de negócio. Tais características da arquitetura J2EE vão ao encontro das necessidades do Direto, e, por isso, foram propostas soluções aos problemas do Direto, baseadas em J2EE, para os módulos de Agenda, Catálogo e Correio.

O JavaMail [Sun Microsystems, 2003b] é o *framework* de correio eletrônico para aplicações Java. Essa API foi projetada visando disponibilizar funcionalidades de correio eletrônico de forma independente de plataformas e protocolos através de uma interface simples e leve, e ao mesmo tempo suportar a criação de interfaces do usuário sofisticadas. Essa API foi utilizada para a proposta de um novo módulo de Correio.

A tecnologia JDO [Sun Microsystems, 2003c], *Java Data Objects*, é uma definição baseada em interface para a persistência de objetos. Essa definição descreve o armazenamento, consulta e recuperação de objetos de bases de dados. Utilizou-se essa tecnologia em conjunto com os Enterprise Java Beans da plataforma J2EE, propondo uma nova forma de persistir os eventos da Agenda de um usuário.

O JBoss [JBoss Group LLC, 2003] é um servidor de aplicações J2EE de código livre, totalmente escrito em Java, porém não certificado oficialmente pela Sun. Em 2002 chegou a 200 mil *downloads* mensais, se tornando o servidor J2EE com maior número de *downloads*, e foi eleito como o melhor servidor de aplicação pela revista online JavaWorld, concorrendo com o BEA, IBM, Sun e outros. O fato do JBoss ser distribuído sob a licença LGPL permite a sua utilização de forma comercial. Por exemplo, é possível distribuir uma aplicação baseada em J2EE em um CD junto com o JBoss, utilizá-lo em conjunto com outras aplicações comerciais, ou até mesmo alterar seu código fonte gerando um novo produto. Além disso, o uso de software gratuito reduz o custo total de propriedade [Gomes, 2002].

3. O Direto e J2EE

Foram propostas quatro modelagens baseadas nas tecnologias apresentadas, para o software Direto. Duas delas são para o módulo Agenda, a primeira baseada em *entity bean*s para persistência do dados [Rosinha, 2002] e a segunda baseada em JDO para persistência [Silveira, 2003]. Também foi proposto um modelo para o módulo de Catálogo, utilizando o protocolo LDAP e também um modelo para o módulo de Correio, utilizando-se *session beans* e a API Java Mail.

Os padrões de projeto de *session beans* de Fachada e de Objetos de Transferência foram amplamente utilizados nos trabalhos e merecem atenção especial.

3.1. Beans de Fachada

Os *session beans* geralmente tem a função de apresentar uma interface unificada a clientes fora do contêiner EJB, para a manipulação de um conjunto de *entity beans*. Para este tipo de função é dado o nome de fachada, por omitir de seu cliente o uso de *entity beans* internos [Maciel, 2002]. O bean de fachada permite o acesso aos serviços oferecidos pelo contêiner EJB. Além disso, como somente ele pode ser acessado remotamente pela camada Web, ele se torna o único ponto de entrada para o modelo da aplicação. Isso resulta em baixo acoplamento entre a interface e o modelo da aplicação [Vidal, 2003].

3.2. Objetos de Transferência

Para a execução dos casos de uso dos modelos que serão apresentados, é necessário que o *session bean* de fachada transmita e/ou receba dados à camada Web de maneira eficiente, usando o menor número de chamadas remotas. Ao mesmo tempo, o *session bean* não deve expor os objetos do

domínio para a camada Web, pois isso prejudicaria a independência entre essas camadas que o *session bean* de fachada garantiria. Para obter uma comunicação dos dados entre as camadas Web e EJB usando uma única chamada remota sem estabelecer dependências entre os objetos da interface e do domínio, os dados são encapsulados em objetos de transferência (*transfer objects*), classes Java que encapsulam os dados a serem transmitidos e são serializáveis, isto é podem ser transformadas em fluxos de dados para comunicação remota [Vidal, 2003].

4. As modelagens Propostas

A seguir são apresentadas as modelagens propostas aos módulos de Agenda, Catálogo e Correio, utilizando-se J2EE em conjunto com outras tecnologias baseadas em software livre.

4.1. O Módulo Agenda

Dois trabalhos foram realizados no intuito de propor modelos J2EE para o módulo de agenda do Direto. O primeiro [Maciel, 2002], baseou-se em persistência com *entity beans* para os eventos de agenda. O segundo [Silveira, 2003], baseou-se em um novo padrão proposto pela SUN para persistência de objetos, o JDO.

Em ambas as modelagens utilizou-se *stateless session beans*, como beans de fachada, pois a análise dos métodos da agenda revelou que a natureza dos métodos expostos é transacional, ou seja, não existe real necessidade de se manter o estado de um *session bean* e mantê-lo alocado exclusivamente para cada usuário acessando o sistema para manter a sua sessão.

Os dois trabalhos realizados para a agenda diferenciam-se em sua camada de persistência. O primeiro trabalho [Rosinha, 2002] utiliza um *entity bean* que representa a entidade evento do domínio do problema. Ele usa persistência *container-managed* (gerenciada pelo contêiner EJB).

Para a modelagem usando-se JDO para persistência dos dados da Agenda [Silveira, 2003], além dos EJBs é necessária a criação de uma classe que será a classe a ser persistida. Essa classe não difere de uma classe Java comum e pelos conceitos JDO é chamada de *PersistenceCapable*, ou seja, uma classe que pode ser persistida. Nesse caso, a utilização de um *entity bean* seria redundante, pois *entity beans* e JDO são utilizados para persistência.

4.2. O Módulo Catálogo

Para a modelagem do catálogo [Maciel, 2002], identificou-se que as entidades que modelam os dados que serão persistidos são grupo e contato e foram modeladas como *entity beans*. A entidade catálogo provê o meio pelo qual os dados de grupo e contato são manipulados, portanto, foi modelada como um *session bean* de fachada.

Para o caso do catálogo, utilizou-se um objeto de transferência para grupo e outro para contato. Quando for requisitado ao bean de fachada os contatos do catálogo pessoal, esse obtém as referências a todos os contatos, encapsula os dados de todos estes objetos em objetos de transferência e os envia para o cliente.

4.3. O Módulo Correio

Embora *entity beans* sejam o componente padrão da especificação J2EE para a modelagem dos objetos do domínio, a modelagem proposta para o módulo de correio não implementa suas classes dessa forma. Esta abordagem, que difere daquelas empregadas nos módulos da Agenda [Rosinha, 2002] [Silveira, 2003] e do Catálogo [Maciel, 2002], deve-se a dois fatores. Primeiro, a persistência gerenciada pelo contêiner que o J2EE oferece é aplicável apenas a banco de dados relacionais; uma modelagem de mensagens e pastas de correio como *entity beans* necessariamente usaria persistência gerenciada pelo bean, eliminando uma das principais vantagens dos

entity beans, que é a persistência transparente gerenciada pelo contêiner. Assim sendo, as classes Mensagens e Pasta não utilizam os recursos dos *entity beans*, podendo ser implementadas como *session beans* ou classes Java simples [Vidal, 2003].

Por outro lado, sendo os objetos de domínio implementados como classes Java simples, eles não tem acesso aos serviços oferecidos pelo contêiner EJB, como controle de transações e de acesso. Para se aproveitar dessas características inerentes aos EJBs, foi necessária a implementação de um *session bean* de fachada para a camada de domínio.

5. Conclusões

A utilização de tecnologias baseadas em software livre é uma ótima escolha para o desenvolvimento de sistemas, pois oferece baixo custo e uma grande liberdade de customização de sistemas de uma empresa. Além disso, apresentou-se algumas soluções baseadas na plataforma J2EE da Sun, para o *software* Direto da PROCERGS, valendo-se da arquitetura em camadas e do fato da tecnologia Java ser livre e poder ser utilizada com tantas outras tecnologias de mesmo filosofia. A arquitetura de camadas imposta pelo J2EE permite a remodelagem do sistema utilizando baixo acoplamento e maior modularidade entre os módulos do Direto.

Referências

DiretoGNU (2003). DiretoGNU - Comunicação Objetiva. Disponível em http://www.direto.org.br.

Geyer, C. F. R. et al. (2001). Projeto Sistemas Avançados para Comunicação Eletrônica - Software Aberto de Correio, Agenda e Catálogo. In *Anais do 2o Workshop sobre Software Livre*, pages 25–28.

Gomes, H. (2002). JBoss 3. *Revista Java Magazine*. Edição 2, Ano 1. Dezembro 2002.

JBoss Group LLC (2003). JBoss. Disponível em: http://www.jboss.org. Acesso em: 06 março 2003.

Maciel, F. R. (2002). Modelagem do Catálogo e Autenticação do Direto utilizando J2EE e JAAS. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Rosinha, R. B. (2002). DiretoJ2EE: Modelagem e Prototipação do Direto para a Plataforma J2EE. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Silveira, K. K. (2003). DiretoJ2EE: Uso de JDO para Persistência no Módulo Agenda. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Sun Microsystems (2003a). J2EE. Disponível em http://java.sun.com/j2ee/index.html. Acesso em: 06 março 2003.

Sun Microsystems (2003b). JavaMail(TM) API v 1.2. Disponível em http://java.sun.com/products/javamail/index.html. Acesso em: 06 março 2003.

Sun Microsystems (2003c). JDO v 1.2. Disponível em http://java.sun.com/products/jdo/index.html. Acesso em: 06 março 2003.

Vidal, S. L. B. (2003). Modelagem J2EE do Módulo Correio do Direto. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.