Prime: uma solução Java para acesso móvel a informações utilizando GSM/GPRS

RENATO BARBOSA MIRANDA
VALESKA PIVOTO PATTA MARCONDES
Inatel - Instituto Nacional de Telecomunicações
ICC - Inatel Competence Center
Av. João de Camargo, 510 CEP:37540-000, Santa Rita do Sapucaí(MG)
(rmiranda, valeska)@inatel.br

Resumo: Aplicações corporativas estão sendo cada vez mais utilizadas pelas empresas para a gestão de negócios, pois permitem que sejam tomadas decisões rápidas, baseadas nas informações mais recentes de sua base de dados. Este artigo tem como objetivo descrever uma solução Java para acesso móvel a informações, apresentando uma arquitetura e ferramentas que suportam tal desenvolvimento.

Nossos resultados demonstram que é possível a utilização da tecnologia Java e a infraestrutura GSM/GPRS, para o acesso móvel às informações corporativas e também a utilização de ferramentas gratuitas de desenvolvimento.

Palavras-Chave: J2ME, J2EE, Wireless, Java, dispositivos móveis.

1. Introdução

Em um mercado competitivo, as empresas necessitam controlar os negócios de forma ágil e eficiente.

Em um ambiente onde as informações não estão totalmente integradas, devido a problemas geográficos ou mesmo por possuir formas diferentes de captação dessas informações, este controle fica prejudicado, tornando-se um fator crítico na gestão do negócio. Fica cada vez mais evidente para as empresas que a necessidade de tomadas de decisões rápidas, porém embasadas em dados reais, é vital para a sobrevivência da mesma no mercado.

Atualmente, algumas tecnologias possibilitam realizar o que, em um passado recente, era inviável: acessar remotamente os dados de uma empresa e poder tomar decisões em tempo real, utilizando para isto um dispositivo móvel.

A evolução da indústria *wireless* através das tecnologias GSM/GPRS , juntamente com a tecnologia Java para o desenvolvimento de aplicações móveis, e dos próprios dispositivos móveis, veio possibilitar a criação de aplicações elaboradas utilizando gráficos, cores, imagens e outros recursos.

O artigo em questão irá expor o trabalho realizado pelo Inatel Competence Center, em parceria com a Motorola (Centro de desenvolvimento de Jaguariúna), no desenvolvimento de uma aplicação corporativa de acesso móvel , permitindo a tomada de decisões baseadas nas informações de produção de uma fábrica.

2. Perspectiva de produto

O sistema Prime foi desenvolvido visando disponibilizar os dados de produção de uma fabrica, contidos em sua base de dados.

Através de um dispositivo móvel GSM/GPRS que suporte Java, o usuário poderá instalar a aplicação, realizar pedidos de produção para a fábrica e acompanhar o andamento dos pedidos existentes.

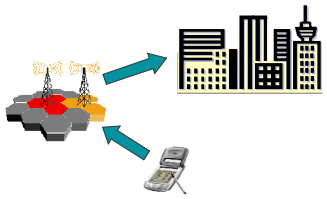

Figura 1 – Perspectiva do produto

3. Visão geral do sistema Prime

As seguintes *use cases* fazem parte do sistema Prime:

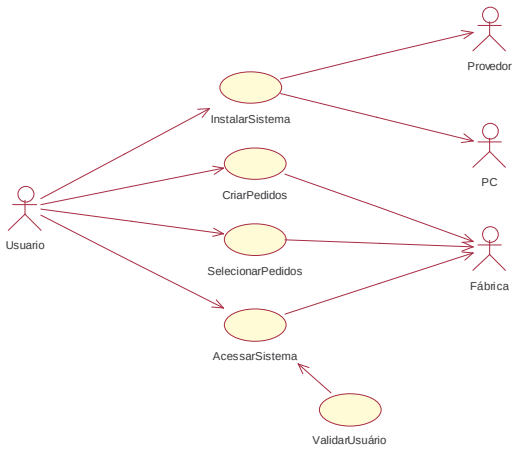

Figura 2 – Diagrama de *Use Cases* do sistema Prime

3.1. Descrição das *use cases*:

- InstalarSistema

Esta *use case* é responsável por instalar o sistema Prime no equipamento que fará o acesso *wireless*.

- AcessarSistema

Esta *use case* é responsável por permitir o acesso ao sistema e garantir que o usuário que disparou a sua execução seja um usuário válido.

- ValidarUsuario

Esta *use case* é responsável por verificar se o nome do usuário e sua senha conferem com os dados armazenados na base de dados da Fábrica.

- CriarPedidos

Esta *use case* é responsável por permitir a criação de pedidos de produção.

- SelecionarPedidos

Esta *use case* é responsável por permitir a seleção de pedidos de produção existentes e apresentar as informações correspondentes.

3.2. Descrição dos atores:

- Usuário

Representa todos os usuários do sistema que podem ser: fornecedores, diretores, gerentes, fabricantes.

- Fábrica

Representa o sistema servidor desenvolvido em J2EE e que possui as informações que serão utilizadas pela aplicação cliente.

- Provedor

Representa o provedor que terá a aplicação disponível para *download* em caso de ser feita a instalação do sistema através de OTA (*Over The Air download*).

- PC

Representa o computador que poderá prover a instalação do sistema através de porta serial ou USB.

4. Especificação de Requisitos

4.1. Mobilidade

O sistema deverá ser utilizado em dispositivos móveis.

4.2. Acesso ao sistema

O sistema deverá prover a segurança de acesso às informações contidas na base de dados, através da validação de usuário e senha.

4.3. Segurança das informações

Cada usuário deverá ter acesso apenas às informações referentes aos seus pedidos de produção.

4.4. Inserir pedidos

O sistema deverá permitir a inserção de pedidos de produção, especificando o produto, a quantidade, a data inicial e a data final de produção.

4.5. Consultar pedidos

O sistema deverá permitir a consulta à informações do andamento dos pedidos de produção. Deverão ser apresentados: a quantidade solicitada, a quantidade já produzida, a porcentagem de erros ocorridos na linha de produção, a data prevista de término e o *status* da produção (em dia, atrasada, adiantada).

4.6. Apresentar gráfico de andamento de produção

O sistema deverá apresentar de forma gráfica, o *status* da produção.

5. Tecnologias adotadas

Para atender os requisitos do sistema Prime, foi escolhida a tecnologia Java e a infra-estrutura de uma operadora GSM/GPRS.

Para o desenvolvimento, foi utilizada a ferramenta Sun One Studio 4 Mobile Edition upgrade 1, da Sun Microsystems.

Como servidor utilizou-se o JBoss versão 3.2.3 e banco de dados MySQL, instalados em uma máquina Sun Solaris.

O dispositivo móvel utilizado para testes e demonstrações do sistema foi o Motorola A388 que é um híbrido de celular com PDA e possui J2ME na versão 1.0.

A seguir será apresentada uma breve descrição da tecnologia Java.

5.1. Visão geral da plataforma JAVA

Java é uma tecnologia criada pela Sun Microsystems que permite o desenvolvimento de programas para serem usados em qualquer plataforma de processamento que possua uma máquina virtual Java (*Java Virtual Machine*).

A primeira edição lançada da plataforma Java foi a *Java Standard Edition* (J2SE), que tem como foco o desenvolvimento de aplicações desktop. Posteriormente foi lançada a edição para servidores *chamada Java Enterprise Edition* (J2EE), que visa o desenvolvimento de aplicações servidoras distribuídas e que necessitam de alta performance.

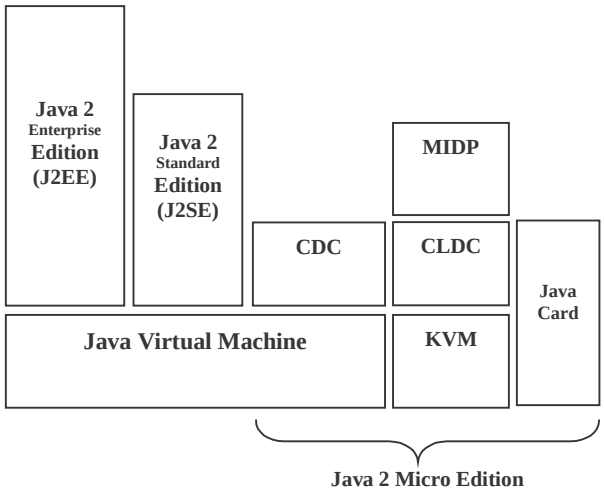

Figura 2 – Visão geral da plataforma Java

Recentemente, tendo em vista o crescente mercado de aplicações embutidas, foi criada a edição *Java Micro Edition* (J2ME), para ser utilizada em dispositivos que abrangem desde TV´s com acesso a Internet até celulares. Atualmente a tecnologia Java conta com mais uma edição chamada JavaCard que permite a criação de aplicações embutidas em cartões para diversas finalidades.

5.2. Java Enterprise Edition (J2EE) – *Server Side*

Essa tecnologia foi criada para possibilitar o desenvolvimento de sistemas servidores que necessitam de um alto desempenho, permitindo o uso de uma arquitetura distribuída.

Permite a utilização de *componentes Enterprise Java Beans* (EJB), que são registrados em um servidor J2EE. Através do servidor, os EJB's disponibilizam suas interfaces que poderão ser utilizadas por uma diversidade de aplicações.

Em um projeto de aplicação distribuída, os EJB de sessão são responsáveis pela camada de *business* do sistema, enquanto os EJB de entidade realizam o acesso aos dados.

Os EJB's podem residir em máquinas distintas possibilitando assim obter-se o máximo desempenho de cada máquina para a realização das transações do sistema.

5.3. Java Micro Edition (J2ME) – *Client Side*

Essa edição da plataforma Java foi construída com o objetivo de ser utilizada em dispositivos de baixa capacidade de processamento e armazenamento tais como: aparelhos celulares, PDAs, Internet *screenphones*, *settop boxes* digitais para TV, sistemas de navegação automotiva, comutadores e roteadores de rede, componentes para automação residencial, etc.

A plataforma J2ME é constituída das seguintes partes:

- KVM – K *Virtual Machine*:

Máquina virtual J2ME para que seja possível a execução das aplicações nos dispositivos móveis.

- CLDC - *Connection Limited Device Configuration*:

Camada de configuração que possui uma API limitada, com o mínimo necessário para que seja executada uma aplicação em um dispositivo móvel de baixa capacidade de processamento.

O CLDC precisa de uma área específica de memória para o seu funcionamento, sendo assim, não há como o desenvolvedor de aplicações acessar partes da memória específica do dispositivo, como agenda de endereços e outras informações que dizem respeito ao usuário do dispositivo, garantindo assim a segurança contra possíveis defeitos de invasão de memória.

Os seguintes pacotes são disponibilizados:

- java.io
- java.lang
- java.util
- javax.microedition.io

Os três primeiros são herdados da edição *Java Standard Edition* (J2SE) porém com menos recursos, como por exemplo: não suportam tipo de dados ponto flutuante, finalização(Object.finalize()) e Java Native Interface (JNI).

- CDC – *Connected Device Configuration*

API utilizada no desenvolvimento de aplicações para dispositivos com maior capacidade de processamento como, por exemplo, PDAs mais robustos.

- MIDP - *Mobile Information Device Profile*

Essa API acrescenta funcionalidades específicas a um determinado tipo de dispositivo móvel. O MIDP define os pacotes que estão no CLDC e adiciona os listados abaixo:

- javax.microedition.lcdui (interface com o usuário)

- javax.microedition.rms (sistema de gerência de registros para persistência de informações)
- javax.microedition.midlet

Atualmente os dispositivos móveis disponíveis no mercado que suportam J2ME utilizam a versão 1.0 do MIDP.

Alguns fabricantes de dispositivos oferecem pacotes adicionais ao MIDP que são compatíveis apenas com determinados modelos. Portanto uma aplicação desenvolvida utilizando esses pacotes adicionais, possuirá operações específicas correspondentes ao modelo do aparelho, o que torna difícil o uso da mesma aplicação para outros modelos com suporte J2ME.

Para que uma aplicação possa ser utilizada em qualquer dispositivo compatível com J2ME, ela deverá ser desenvolvida apenas utilizando os pacotes existentes na versão MIDP suportada.

Figura 3 - Alguns dispositivos que suportam J2ME

6. Arquitetura do sistema:

O sistema possui três camadas:

- Camada cliente: representada pela aplicação J2ME carregada dentro de um dispositivo móvel.
- Camada servidora de aplicação: é constituída por:
  - EJB's de sessão responsáveis pela lógica de negócios.
  - Scripts JSP (Java Server Pages), responsáveis por controlar a conexão HTTP com a aplicação J2ME.
  - Servidor J2EE responsável por prover a infra-estrutura para uso dos EJB's.
- Camada servidora de dados: é constituída por:
  - EJB's de entidade, responsáveis pelo acesso aos dados da base de dados.
  - Banco de dados.

A figura 4 a seguir ilustra como está elaborada a arquitetura do sistema Prime.

A seguir serão apresentados os passos de 1 a 8 para que a aplicação J2ME acesse os dados contidos na base de dados da fábrica para que sejam validadas as informações de nome e senha do usuário do dispositivo móvel:

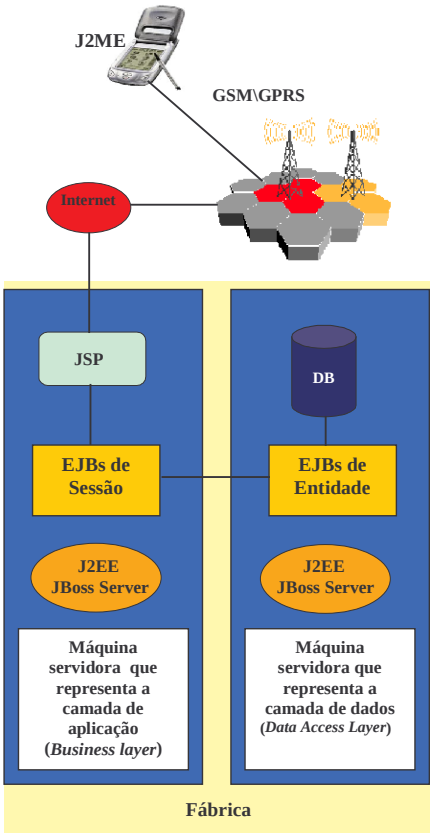

Figura 4 - Arquitetura sistema Prime

1. O usuário preenche os campos da tela de login com nome de usuário e senha (Figura 5):

Figura 5 - Tela de Login do sistema Prime

2. Ao escolher a opção "Ok", a aplicação irá criar uma conexão com o servidor da camada de aplicação utilizando a classe HttpConnection (J2ME):

```
//Cria a conexão
HttpConnection m_httpConnection = (HttpConnection)
Connector.open("Http://www4.inatel.br:8080/login.jsp",
Connector.READ_WRITE);
// Configura conexão
m_httpConnection.setRequestMethod(HttpConnection.P
OST);
```

```
m_httpConnection.setRequestProperty("Content-type",
"application/x-www-form-urlencoded");
```

3. Utilizando a conexão criada, a aplicação J2ME enviará para o JSP responsável pelo processamento de login, os parâmetros digitados pelo usuário:

```
// Cria um OutputStream para enviar os parâmetros
outputData = m_httpConnection.openOutputStream();
// Cria um vetor com os parâmetros que serão enviados
Vector m_dataVector = new Vector();
m_dataVector.addElement("user=" + m_strUsername);
m_dataVector.addElement("&pass=" + m_strPassword);
// envia os parâmetros que estão dentro do vetor
for(int iCounter = 0; iCounter < m_dataVector.size();
iCounter++) {
// Passa os parâmetros para bytes
String m_data = new
String((String)m_dataVector.elementAt(iCounter)).getB
ytes();
// envia o parâmetro
        outputData.write(m_data);
        m_data = null;
}
```

Neste momento a aplicação J2ME fica em espera de um retorno do servidor.

4. O JSP (login.jsp da camada de aplicação) receberá os parâmetros enviados:

```
<%
   // Recebe o parâmetro com o nome de usuário
   String strUserName = request.getParameter("user");
   // Recebe o parâmetro com a senha
   String strPassword = request.getParameter("pass");
   // Configura o modo de retorno do JSP para aplicação
   // J2ME
   response.setContentType("text/plain");    %>
```

5. O JSP (login.jsp) criará o EJB correspondente por validar o usuário do sistema:

```
<%
ICCUserValidateHome iccUserValidateHome = null;
ICCUserValidate iccUserValidate = null;
try {
  InitialContext ctxInitialContext = null;
  Object objObject = null;
  Hashtable env = new Hashtable();
  env.put(Context.INITIAL_CONTEXT_FACTORY,
  "org.jnp.interfaces.NamingContextFactory");
  env.put(Context.PROVIDER_URL, "localhost:1099");
  env.put(Context.URL_PKG_PREFIXES,
  "org.jboss.naming:org.jnp.interfaces" );
  ctxInitialContext = new InitialContext(env);
  objObject =
ctxInitialContext.lookup("ejb/UserValidate
  JNDI");
iccUserValidateHome = (ICCUserValidateHome)
PortableRemoteObject.narrow(objObject,
ICCUserValidateHome.class);
iccUserValidate = iccUserValidateHome.create
(strUserName);
```

Portanto agora o JSP tem uma instancia do EJB de sessão (iccUserValidate) que irá fazer a validação de usuário, utilizando para isso o EJB de entidade correspondente à tabela de usuários do banco de dados. OBS: Não será abordado como o EJB de sessão utiliza o EJB de entidade.

6. O JSP (login.jsp) utilizará da interface disponibilizada pelo EJB(iccUserValidate) para fazer a validação do usuário e retornar o status dessa para a aplicação J2ME do dispositivo móvel:

```
// Verifica se EJB criado não está com valor nulo
if(iccUserValidate != null)
{
     try {
     // Verifica se password é válida, utilizando interface
     // verifyPassword do EJB instanciado
     if(iccUserValidate.verifyPassword(strPassword)) {
       // envia de volta para a aplicação J2ME
       out.write("valido");
      } else {
        // envia de volta para a aplicação J2ME
        out.write("invalido");
     }
}
```

7. A aplicação J2ME do dispositivo móvel que estava em espera, irá receber e verificar a informação retornada pelo JSP:

```
// Cria um InputStream que irá receber o que foi enviado
// pelo JSP(login.jsp)
InputStream inputData =
m_httpConnection.openInputStream();
// Verifica o código padrão de retorno da conexão Http
if(m_httpConnection.getResponseCode()
=HttpConnection.HTTP_OK) {
    long lSize = -1;
    // Verifica a quantidade de informação que foi
    // retornada
    lSize = m_httpConnection.getLength();
    if(lSize != -1) {
       m_data = new byte[(int)lSize];
       if(inputData != null) {
         inputData.read(m_data);
         // Armazena o que recebeu em uma string
         m_strResultOfRequest = new String(m_data);
       }
    }
```

}

8. A aplicação J2ME terá em m_strResultOfRequest a informação que foi retornada pelo JSP e, através de uma verificação do valor, poderá mostrar a tela de menu do sistema ou uma tela contendo a mensagem do erro ocorrido.

Sempre que a aplicação J2ME fizer uma requisição ao servidor, esses passos serão seguidos, com a diferença que será acessado o JSP correspondente à tarefa que será executada.

7. Formas de se carregar uma aplicação J2ME em um dispositivo móvel

A aplicação poderá ser carregada no dispositivo de três formas:

- OTA (Over the Air download)

Utilizando esse meio é possível fazer o download de uma aplicação J2ME, sendo que para isso a aplicação deverá estar disponível em um servidor Web.

- Infravermelho

Utilizando esse meio é possível carregar uma aplicação no dispositivo através de uma sincronização de dados infravermelho com outro dispositivo que contenha a aplicação.

- Cabo de dados (Serial ou USB)

Esse modo permite que a aplicação J2ME seja carregada no dispositivo através de um programa de sincronização proprietário, instalado em um computador que, a partir de uma conexão serial ou USB, irá fazer a transferência da aplicação existente no computador para o dispositivo.

As formas para se carregar a aplicação em um dispositivo são dependentes das características de cada aparelho.

8. Conclusões e futuros trabalhos

Utilizando as tecnologias demonstradas, foi possível a implementação do sistema Prime, atendendo a todos os seus requisitos.

A tecnologia Java demonstrou ser viável, uma vez que a plataforma Java Enterprise Edition (J2EE) permite o desenvolvimento de aplicações servidoras simples, portáveis e possíveis de se integrar com outros sistemas existentes.

A versão 1.0 do Java Micro Edition (J2ME) possui limitações quanto do desenvolvimento de interfaces gráficas mais elaboradas.

O dispositivo móvel A388 atendeu muito bem às necessidades dos sistema Prime, por se tratar de um dispositivo com características de um PDA, ou seja, com satisfatória capacidade de processamento e armazenamento, e maior facilidade de digitação de dados.

Os testes da aplicação foram realizados utilizando a operadora TIM Campinas e demonstraram um tempo de acesso satisfatório.

Como trabalho futuro, a aplicação poderá ser melhorada utilizando a nova versão da edição J2ME, que consiste no *upgrade* do CLDC 1.0 para CLDC 1.1 e MIDP 1.0 para MIDP 2.0. Entre as novidades da nova versão do MIDP destacam-se:

- API's para desenvolvimento de interfaces gráficas mais elaboradas
- Criação de conexões HTTPS
- MMA, Mobile Multimedia API, que permite o uso de áudio e vídeo em dispositivos móveis.

Também poderão ser acrescentados novos requisitos no sistema Prime, como o cancelamento de pedidos de produção e a utilização de criptografia de dados.

9. Glossário:

- API – Application Program Interface
- HTTP – Hypertext Tranfer Protocol
- HTTPS - Hypertext Tranfer Protocol Securiy
- JSP – Java server Pages
- USB – Universal Serial Bus
- *Use Cases* = conjunto de funcionalidades de um sistema, representado por fluxos de eventos iniciados por um ator e apresentando um resultado de valor a um ator.
- Ator – qualquer pessoa ou sistema externo que tenha interação com o sistema em desenvolvimento.

10. Bibliografia

http://java.sun.com/j2ee

http://java.sun.com/j2me

http://www.soujava.org.br

http://www.gsmworl.com

http://www.motorola.com

Livros:

Core J2ME Technology e MIDP, John W. Muchow, Sun Microsystems.

Desenvolvendo aplicações comerciais em Java com J2EE e UML, Khawar Zaman Ahmed e Cary E. Umrysh, editora Ciência Moderna.

UML na prática – do problema ao sistema, Caique Cardoso, editora Ciência Moderna.