**UNIVERSIDADE REGIONAL DE BLUMENAU**

**CENTRO DE CIÊNCIAS EXATAS E NATURAIS**

**CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO**

# PROTÓTIPO DE SOFTWARE PARA GERENCIAMENTO DE SERVIDORES LINUX ATRAVÉS DE DISPOSITIVOS MÓVEIS COM SISTEMA OPERACIONAL SYMBIAN S60

**RAFAEL SCHLEUSS**

**BLUMENAU**
**2010**

**2010/1-20**

**RAFAEL SCHLEUSS**

# PROTÓTIPO DE SOFTWARE PARA GERENCIAMENTO DE SERVIDORES LINUX ATRAVÉS DE DISPOSITIVOS MÓVEIS COM SISTEMA OPERACIONAL SYMBIAN S60

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Francisco Adell Péricas - Orientador

**BLUMENAU**
**2010**

**2010/1-20**

# PROTÓTIPO DE SOFTWARE PARA GERENCIAMENTO DE SERVIDORES LINUX ATRAVÉS DE DISPOSITIVOS MÓVEIS COM SISTEMA OPERACIONAL SYMBIAN S60

Por

**RAFAEL SCHLEUSS**

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

| | |
|---|---|
| Presidente: | Prof. Francisco Adell Péricas – Orientador, FURB |
| Membro: | Prof. Paulo Fernando da Silva – FURB |
| Membro: | Prof. Sérgio Stringari – FURB |

Blumenau, 1 de julho de 2010

Dedico este trabalho a minha esposa que por todo este tempo esteve ao meu lado, me dando forças para continuar. E também ao meu avô, que me possibilitou o começo desta grande caminhada.

## AGRADECIMENTOS

A minha esposa Angela pela força e persistência em me manter no caminho certo.

Aos meus amigos, por acreditarem, ou não, que um dia eu chegaria lá.

Ao meu orientador, Francisco Adell Péricas pela força na realização deste trabalho.

A imaginação é mais importante que o conhecimento.

Albert Einstein

# RESUMO

Este trabalho apresenta um estudo sobre gerenciamento de servidores Linux e o desenvolvimento de aplicativos para dispositivos móveis, objetivando a especificação e implementação de um protótipo de software para monitoração e configuração de servidores Linux através de um dispositivo portátil com Sistema Operacional Symbian S60.

Palavras-chave: Symbian. J2ME. Gerenciamento de redes. Linux.

# ABSTRACT

This paper presents a study about management of Linux servers and application development for mobile devices, aiming at the specification and implementation of a prototype for monitoring and configuration of Linux servers through a portable device with Operational System Symbian S60.

Key-words: Symbian. J2ME. Network management. Linux.

# LISTA DE ILUSTRAÇÕES

# LISTA DE SIGLAS

*API – Application programming interface*

BSD – *Berkeley Software Distribution*

*CDC – Connected Device Configuration*

*CLDC – Connected Limited Device Configuration*

CPU – *Central Processing Unit*

ISO – *International Organization for Standardization*

J2ME – *Java 2 Micro Edition*

*JSP – Java Server Pages*

LSB – *Linux Standard Base*

*MIDP – Mobile Information Device Profile*

PDA – *Personal Digital Assistant*

SQL – *Structured Query Language*

SSH – *Secure Shell*

SVG – *Scalable Vector Graphics*

# SUMÁRIO

# 1 INTRODUÇÃO

A área de gerência de redes foi inicialmente impulsionada pela necessidade de monitoração e controle do universo de dispositivos que compõem as redes de comunicação (REDE NACIONAL DE ENSINO E PESQUISA, 1997), sendo geralmente utilizado um computador de mesa ou um *notebook* para tal finalidade.

A gerência de redes de computadores é uma tarefa complexa, em boa parte por consequência do crescimento acelerado das mesmas, tanto em desempenho quanto em suporte a um grande conjunto de serviços. A gerência está associada ao controle de atividades e ao monitoramento do uso de recursos da rede. As tarefas básicas da gerência de redes, simplificadamente, são obter informações da rede, tratar estas informações, possibilitando um diagnóstico, e encaminhar as soluções dos problemas (SZTAJNBERG, 1996).

No sistema operacional Linux várias ferramentas para gerenciamento e monitoração já estão presentes em uma instalação padrão, as quais facilitam ao administrador de sistemas que possua inclusive um terminal remoto. Em casos em que o administrador não esteja presente na empresa, ou mesmo longe de sua mesa, pode haver uma demora para a detecção de um problema ou uma manutenção.

Para Pretto et al. (2006), tecnologias de computação e comunicação móvel estão em contínuo avanço em termos de disponibilidade, funcionalidade e custos, tornando-se atraentes para os planos de automatização das empresas. Segundo Moreira (2006), estes dispositivos disponibilizam algumas das ferramentas e recursos presentes apenas em um computador pessoal, suprindo necessidades relacionadas ao lazer e ao trabalho.

Muito importante também para a popularização da tecnologia móvel são os sistemas operacionais que trazem cada vez mais funcionalidades para os usuários da tecnologia móvel. Presente em sessenta por cento dos aparelhos do segmento de *smartphones* (MARTIN, 2008) está o sistema operacional Symbian.

Muito popular no Brasil e Europa, presente em celulares da Nokia, Samsung e LG, o Symbian consegue atender usuários tanto corporativos quanto os que buscam um aparelho com mais entretenimento (COVERT; BURGOS, 2009). Devido ao fato de permitir o desenvolvimento de softwares em linguagens populares como o Java, Python e C++, e de possuir rica documentação, ambiente de desenvolvimento e ambiente de simulação, o Symbian possui muitos adeptos e uma grande quantidade de softwares disponíveis.

Diante do exposto, pretende-se utilizar a plataforma Symbian S60 como base para o

desenvolvimento de um protótipo de software para o gerenciamento e monitoração dos recursos de servidores através de um *smartphone* ou celular

## 1.1 MOTIVAÇÃO

Segundo Kurose; Ross (2006, p. 571), o administrador de rede exerce a tarefa de manter a rede "viva e atuante", e, portanto, deve estar habilitado a reagir a contratempos no menor tempo possível.

Dentro das pequenas empresas, a pessoa que gerencia a rede normalmente exerce também outras atividades. Com isso o trabalho de realizar pequenas modificações como a liberação de internet aos usuários internos, ou mesmo o monitoramento do espaço em disco, se torna mais demorado. A utilização do celular, cada vez mais presente na vida das pessoas, nestas tarefas triviais pode facilitar e agilizar o trabalho do administrador da rede.

## 1.2 OBJETIVOS DO TRABALHO

O objetivo deste trabalho foi desenvolver uma interface remota para gerenciamento e monitoração de servidores Linux utilizando dispositivos móveis dotados de sistema operacional Symbian.

Os objetivos específicos do trabalho são:

a) efetuar o gerenciamento dos usuários do servidor;
b) gerenciar o acesso aos recursos de internet e intranet;
c) permitir a monitoração do tráfego de rede;
d) permitir a monitoração dos recursos do sistema operacional;
e) efetuar alertas para o administrador em caso de problemas.

## 1.3 ESTRUTURA DO TRABALHO

O trabalho está organizado em seis capítulos, estando assim divididos:

O primeiro capítulo apresenta uma introdução sobre o assunto, os objetivos e a organização do trabalho.

O segundo capítulo apresenta os conceitos e objetivos da gerência de redes.

O terceiro capítulo é dedicado a apresentação do sistema operacional Linux e sua utilização como *firewall* de uma rede.

O quarto capítulo aborda os disposivos móveis, a utilização do Symbian, como sistema operacinal para celulares e *handhelds*, a plataforma Java 2 Micro Edition (J2ME) e a utilização de Midlets.

O quinto capítulo abrange o desenvolvimento do trabalho. As ferramentas e bibliotecas utilizadas no desenvolvimento, bem como a especificação do protótipo.

O sexto capítulo conclui o trabalho, apresenta limitações e também, sugestões para trabalhos futuros.

# 2 GERENCIAMENTO DE REDES

Segundo Carmona; Hexsel (2005, p. 324), qualquer atividade de administração, correção de erros do sistema ou assistência para resolução de problemas que não seja realizada de forma direta, ou seja, em que o profissional não lide pessoalmente com a instalação física do servidor ou estação de trabalho com problemas, é considerada um tipo de administração remota. O administrador de rede, cuja tarefa é manter a rede "viva e atuante", deve estar habilitado a reagir a contratempos (e, melhor ainda, a evitá-los) (KUROSE; ROSS, 2006, p. 571).

Conforme Kurose e Ross (2006, p. 573), a *International Organization for Standardization* (ISO) criou um modelo de gerenciamento de rede onde são definidas cinco grandes áreas:

a) gerenciamento de desempenho. A meta do gerenciamento de desempenho é quantificar, medir, informar, analisar e controlar o desempenho de diferentes componentes da rede;
b) gerenciamento de falhas. O objetivo do gerenciamento de falhas é registrar, detectar e reagir às condições de falha da rede;
c) gerenciamento de configuração. O gerenciamento de configuração permite que o administrador saiba quais dispositivos fazem parte da rede administrada e quais são as suas configurações de hardware e software;
d) gerenciamento de contabilização. O gerenciamento de contabilização permite que o administrador da rede especifique, registre e controle o acesso de usuários e dispositivos aos recursos da rede. Quotas de utilização, cobrança por utilização e alocação de acesso privilegiado a recursos fazem parte do gerenciamento de contabilização;
e) gerenciamento de segurança. A meta do gerenciamento de segurança é controlar o acesso aos recursos da rede de acordo com uma política definida.

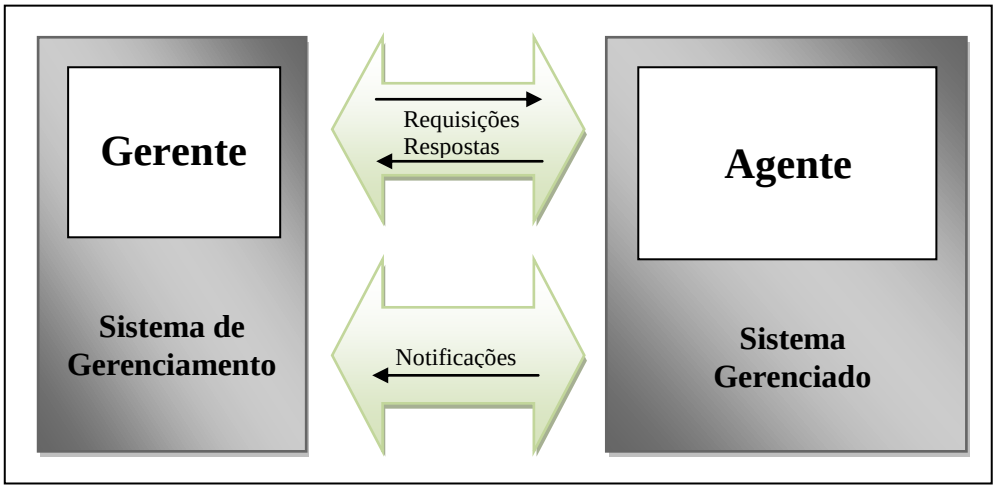
Atualmente as redes de computadores e os seus recursos associados, além das aplicações distribuídas, têm se tornado fundamental e de tal importância para uma organização, que elas basicamente não podem falhar (REDE NACIONAL DE ENSINO E PESQUISA, 1997).

Ainda, conforme Rede Nacional de Ensino e Pesquisa (1997), isso significa que o nível de falhas e de degradação de desempenho considerados aceitáveis está cada vez mais

diminuindo, sendo este nível igual a zero, dependendo da importância da rede para uma instituição.

Conforme Péricas (2003, p. 122), gerência de segurança permite prevenir e detectar o uso impróprio ou não autorizado de recursos de uma rede, assim como administrar a sua segurança. Para realizar esta tarefa, os administradores de rede são geralmente auxiliados por um sistema de gerência de redes que por sua vez pode ser definido como um conjunto de ferramentas integradas para a monitoração e controle. Este sistema pode apresentar uma interface única, com informações sobre a rede, e pode oferecer também um pacote de comandos que são utilizados para executar quase todas as tarefas de gerência de rede.

Uma aplicação de gerência no papel de gerente faz requisições de operações e recebe notificações, enquanto que o agente processa estas operações, envia as respostas e emite as notificações (PÉRICAS, 2003, p. 125). Na Figura 1 - Gerência de redes pode-se ver um exemplo de comunicação entre as aplicações gerente e agente.




Fonte: Péricas (2003, p. 125)

Figura 1 - Gerência de redes

Conforme Lopes, Sauvé e Nicolletti (2003, p. 4), a arquitetura geral dos sistemas de gerência de redes apresenta quatro componentes básicos: os elementos gerenciados, as estações de gerência, os protocolos de gerência e as informações de gerência.

Exemplos de elementos gerenciados incluem:

a) hardware: equipamentos de interconexão, enlaces de comunicação, hospedeiros, *nobreaks*, modems, impressoras etc;
b) software: sistemas operacionais, servidores de bancos de dados, servidores de internet, servidores de correio eletrônico etc.

Em um sistema de gerência de redes deve haver pelo menos uma estação de gerência. As estações de gerência são hospedeiros munidos de software necessário para gerenciar a

rede. Para facilitar a vida dos especialistas em gerência, as estações de gerência são normalmente centralizadas, aliás, é muito frequente que haja uma única estação de gerência.

# 3 SISTEMA OPERACIONAL LINUX

Conforme Nemeth, Snyder e Hein (2007, p. 4) o Linux se originou em 1991 como um projeto pessoal de Linus Torvalds, um universitário finlandês. Este projeto foi originalmente concebido como uma modesta ramificação do Minix, um sistema operacional modelo escrito por Andrew S. Tannenbaum.

Ainda segundo Nemeth, Snyder e Hein (2007, p. 5) o Linux difere de outras variantes Unix pelo fato de que o projeto do *kernel* básico define apenas um *kernel* de sistema operacional. O *kernel* tem que ser empacotado junto com comandos, *daemons* e outros softwares para assim formar um sistema operacional completo e utilizável, comumente conhecido como distribuição.

As distribuições variam em seu foco, suporte e popularidade. O Quadro 1 - Lista de distribuições Linux apresenta as distribuições de uso geral mais populares.

| Nome | Site web | Comentários |
|---|---|---|
| CentOS | www.centos.org | Correspondente livre/gratuito do Red Hat Enterprise Linux |
| Debian | www.debian.org | Uma distribuição não comercial popular |
| Fedora | www.fedoraproject.org | Red Hat descorporatizado |
| Gentoo | www.gentoo.org | Distribuição baseada em código fonte |
| Mandriva | www.mandriva.com | Uma das distribuições mais amigáveis ao usuário |
| openSuse | www.opensuse.org | Correspondente livre/gratuito do SUSE Linux Enterprise |
| Red Hat Enterprise | www.redhat.com | Red Hat supercorporatizado |
| Slackware | www.slackware.com | Distribuição esquelética, básica e estável. |
| SUSE Linux Enterprise | www.novel.com/linux | Forte na Europa, em vários idiomas |
| TurboLinux | www.turbolinux.com | Forte na Ásia, em vários idiomas |
| Ubuntu | www.ubuntu.com | A versão simplificada do Debian |

Fonte: Nemeth, Snyder e Hein (2007)

Quadro 1 - Lista de distribuições Linux

Segundo Campos (2002), o desenvolvimento da comunidade Linux, a entrada de participantes de maior porte no mercado criado em torno deste sistema operacional e o surgimento de interesse pela distribuição de aplicações e serviços baseados no Linux, criaram a necessidade de convergência em torno de um padrão, sem restringir a possibilidade de diferenciação entre as distribuições, mantendo assim a liberdade, sem permitir a fragmentação do Linux em uma série de alternativas incompatíveis entre si.

Ainda segundo Campos (2002), esta necessidade levou à criação do Linux Standard Base (LSB), que visa desenvolver e promover um conjunto de padrões para aumentar a compatibilidade entre as distribuições de Linux, permitindo assim que softwares aplicativos possam rodar em qualquer sistema em conformidade com o padrão. Com esta padronização do sistema, pode-se ampliar a compatibilidade de uma aplicação no sistema operacional Linux.

## 3.1 CARACTERÍSTICAS

Segundo Ball; Duff (2004, p. XXXIII), usar Linux é uma boa idéia por vários motivos, que incluem:

a) o Linux fornece um excelente retorno sobre o investimento. Há pouco ou nenhum custo ‘por máquina’;
b) o Linux funciona bem no *desktop*;
c) o Linux pode funcionar como uma plataforma de servidor. Ele é rápido, estável, escalável e robusto.

As pequenas empresas percebem economias adicionais com o Linux porque praticamente todo o software e atualizações exigidas são gratuitos. Servidores Linux podem funcionar como uma solução parcial inicial ou imitar servidores de arquivo, correio ou impressão de outros sistemas operacionais (BALL; DUFF, 2004, p. XXXIII). Com isso, pequenas empresas podem se beneficiar com uma solução com baixo custo de implantação para seus servidores de arquivos e o gerenciamento de acesso a internet por conta de seus funcionários.

## 3.2 COLETANDO INFORMAÇÕES DO SERVIDOR

O *kernel* do Linux possui duas funções fundamentais: controlar o acesso aos dispositivos e gerenciar quando e como os processos acessam estes dispositivos. O sistema de arquivos /proc é uma hierarquia de arquivos especiais que representam o estado atual do *kernel*, permitindo que usuários e aplicações consultem estas informações (RED HAT

LINUX, 2003). Alem disso, alguns dos arquivos deste diretório podem ser manipulados pelos usuários e aplicações para comunicar alterações de configuração ao *kernel*.

Infelizmente, não é possível gravar em todos os arquivos (independentemente das suas aparentes permissões) e não há muita documentação disponível. Alguns valores e seus significados podem ser encontrados na própria documentação do *kernel* (NEMETH; SNYDER; HEIN, 2007, p. 47).

No Quadro 2 é apresentado o comando para alterar o número máximo de arquivos que o sistema pode abrir de uma vez.

```
#echo 32768 > /proc/sys/fs/file-max
```

Fonte: Nemeth, Snyder e Hein (2007)

Quadro 2 – Alterando o número máximo de arquivos abertos

Esse sistema torna fácil a consulta e alteração das configurações do sistema, porém todas as mudanças são perdidas após a reinicialização.

No Quadro 3 são apresentados alguns arquivos de informações existentes no diretório /proc.

| Arquivo | Descrição |
|---|---|
| /proc/cpuinfo | Informações sobre o CPU |
| /proc/apm | Informações sobre o gerenciamento de energia |
| /proc/iomem | Informações sobre o mapa da memória do sistema |
| /proc/loadavg | Informações sobre a carga do processador |
| /proc/meminfo | Informações sobre uso da memória do sistema (Física e Swap) |
| /proc/stat | Informações sobre a utilização do CPU |

Fonte: Nemeth, Snyder e Hein (2007)

Quadro 3 – Arquivos de informações do kernel

Segundo Nemeth, Snyder e Hein (2007, p. 47), comandos como os **ps** e **top** utilizam as informações contidas neste sistema de arquivos para apresentar o *status* dos processos em execução. Utilizando o comando **vmstat** ou simplesmente consultando o conteúdo do arquivo /proc/stat o administrador pode consultar informações sobre a utilização do processador. O Quadro 4 mostra a consulta desta informação:

```
[root@firewall ~]# cat /proc/stat
cpu  8172607 515741 1188234 97588984 3212195 204899 308902 0
cpu0 8172607 515741 1188234 97588984 3212195 204899 308902 0
intr 1312367273 1112176409 8 0 1 4 1 1 2 1 1 1 1 106 0 4891737
18051764 0 0 0 0                                        ctxt 178697081
btime 1272751868
processes 385159
procs_running 1
procs_blocked 0
```

Quadro 4 – Exemplo do conteúdo do arquivo /proc/stat

A primeira linha do resultado apresenta a quantidade de tempo de processamento gasto para realizar diferentes tipos de tarefas. O significado de cada coluna apresentada é:

a) `user`: processos executando no modo usuário;
b) `nice`: processos no modo usuário com prioridade de execução alterada;
c) `system`: processos executando no modo kernel;
d) `idle`: tempo de espera;
e) `iowait`: tempo de espera para tarefas de entrada e saída;
f) `irq`: serviço de interrupções;
g) `softirq`: serviço de *Interrupt Request Line* (IRQ).

## 3.3 FIREWALL

Segundo Cheswick, Bellovin e Rubin (2005, p. 177), *firewall* é qualquer dispositivo, software ou equipamento que limita o acesso a uma rede. O *firewall* é composto pelas seguintes propriedades:

a) todo tráfego de dentro para fora de uma rede, e vice-versa, deve passar pelo *firewall*;
b) apenas tráfego autorizado, definido pela política de segurança local, terá permissão de passar;
c) o próprio *firewall* é imune a penetrações.

Ainda segundo Cheswick, Bellovin e Rubin (2005, p. 177), uma falha em qualquer um destes aspectos não invalida o *firewall*, porém o torna menos seguro.

Conforme Neto (2004, p. 6), um *firewall* atuando como um ponto de indução, ou seja, sendo o único computador diretamente conectado a internet, poderá de forma segura levar serviços de interconectividade à rede local, evitando assim que cada host de sua *Local Area Network* (LAN) seja responsável por sua segurança.

O papel fundamental de um *firewall* é monitorar o tráfego da rede e bloquear o acesso baseado em regras definidas pelo administrador. Algumas regras implementadas permitem o bloqueio de conexões para determinados protocolos e portas.

De acordo com a NBSO (2003), um *firewall* bem configurado é um instrumento importante para implantar a política de segurança da rede, porém os *firewalls* não são infalíveis. A simples instalação de um *firewall* não garante que sua rede esteja segura contra

invasores.

Segundo Cheswick, Bellovin e Rubin (2005, p. 33), mesmo que um *firewall* fosse impermeável e mesmo que os administradores não cometessem equívocos, a internet não pode ser considerada a única fonte de riscos. Em muitos casos a ameaça à rede interna pode partir de dentro da própria empresa.

Por este motivo o papel da gerência de redes é tão importante. Quanto mais cedo o administrador percebe um eventual problema ou ataque, com mais rapidez poderá agir e solucionar o mesmo.

# 4 DISPOSITIVOS MÓVEIS

Segundo Laudon e Laudon (1999, p. 160), novas formas de comunicação estão sendo proporcionadas por redes móveis de dados, usando dispositivos móveis, como telefones celulares e *Personal Digital Assistant* (PDA). Não apenas celulares ou agendas eletrônicas, os dispositivos móveis são pequenos computadores que podem ser facilmente levados a qualquer lugar, criados para atender profissionais e pessoas em movimento que necessitam de rapidez, facilidade e segurança no acesso a informações corporativas e pessoais. Além disso, a tecnologia *wireless* fez com que a indústria deste setor tenha tido um crescimento explosivo nos últimos anos, permitindo que as pessoas comuniquem-se de forma barata e fácil sem ficarem presas aos seus telefones ou computadores de mesa

Os dispositivos oferecem muitos recursos, entre eles o ambiente Java, que possibilita o desenvolvimento de inúmeras aplicações para auxiliar na coleta e análise de informações e na tomada de decisões. Segundo Rezende; Abreu (2001, p. 88), os dispositivos móveis podem apresentar recursos hoje em dia essenciais para todas as empresas.

Para aqueles que consomem grande parte do seu tempo trabalhando remotamente, estes equipamentos são versáteis, dedicados, multifuncionais e de uso genérico. Do ponto de vista empresarial, eles são ótimos geradores de informações, podendo ser utilizados desde na automação de processos até na coleta de informações estratégicas.

## 4.1 SISTEMA OPERACIONAL SYMBIAN

A história do Symbian começa em 1984 quando uma pequena empresa Inglesa denominada Psion começou a produzir computadores de mão. O Psion Series 5, como era chamado, era um *handheld* bastante poderoso (para a época), que oferecia um volume surpreendente de recursos e rodava um sistema operacional próprio, o EPOC (MORIMOTO, 2008). Este aparelho chamou a atenção dos fabricantes de dispositivos celulares, que já vislumbravam a tendência destes aparelhos em agregar novas tecnologias e funcionalidades (IIDA, 2006, p. 14).

Em termos de hardware, o Psion 5 era bastante modesto, a começar pelo processador. O grande destaque era o sistema operacional, que aproveitava muito bem os recursos de

hardware, e rodava com um desempenho surpreendente (MORIMOTO, 2008).

Porém, ainda segundo Morimoto (2008), nenhum dos dispositivos da Psion fez muito sucesso, o que acabou levando a empresa a descontinuar a linha em 2001. Apesar disso, o EPOC sobreviveu, dando origem ao Symbian, o sistema mais usado em smartphones, sobretudo em aparelhos da Nokia, LG, Samsung, Motorola e Sony-Ericsson.

Duas vantagens do Symbian são que ele tem o apoio de vários fabricantes (incluindo a Nokia, que possui mais de 30% mercado global de aparelhos) e que ele é um sistema relativamente leve (principalmente se comparado ao Windows Mobile, que seria seu concorrente direto), permitindo que ele seja usado em aparelhos mais compactos, sem que as funções ou a autonomia da bateria sejam comprometidas (MORIMOTO, 2008).

Symbian é um sistema operacional de 32bits, multitarefa e robusto, desenhado especialmente para o ambiente *wireless* e para as restrições dos telefones celulares.

Ele foi baseado em cinco pontos chaves: pequenos dispositivos móveis, mercado de massa, conexão *wireless* ocasional, variedade de produtos e plataforma aberta para desenvolvimento de terceiros (DELALANDE, 2003).

Como resultado dos esforços da *Open Mobile Alliance* 5 (OMA), o Symbian é o sistema operacional móvel de maior participação no mercado, com seu maior enfoque na Europa (AZAMBUJA, 2007). Ainda conforme Azambuja (2007), outra importante peculiaridade do sistema, inerente ao conceito de *Open Standard Operating System* (Sistema Operacional Aberto Padrão) de qual faz parte, se traduz na possibilidade de customização e personalização do sistema e de seus aplicativos por parte dos desenvolvedores, da operadora ou do próprio usuário.

Segundo VivaSemFio (2008), na área do desenvolvimento de aplicações o Symbian é um sistema operacional muito versátil, permitindo o desenvolvimento de aplicativos em diversas linguagens. Como exemplo, há o Symbian C/C++, JavaME, FlashLite, Perl, Python, Ruby e até Lua (criada em 1993 na Pontifícia Universidade Católica do Rio de Janeiro, no Brasil).

Ainda segundo VivaSemFio (2008), a versão mais recente do Symbian é a 9.5. Ela tem suporte a banco de dados SQL e a TV Digital nos padrões DVB-H e ISDB-T. Melhorou os recursos de câmera, serviços de localização e transferência entre redes Wi-Fi e 3G. Também reduziu o consumo de memória em 25%, permitindo ao processador entregar mais resultados num intervalo de tempo menor.

## 4.2 JAVA

Segundo Muchow (2004), o Java surgiu da idéia de desenvolver uma linguagem de programação na qual o desenvolvedor escreve o código uma vez e executa em qualquer plataforma que suporte uma máquina virtual Java.

Segundo Moraes (2009, p. 2), desde seu lançamento, em maio de 1995, a plataforma Java foi adotada mais rapidamente do que qualquer outra linguagem de programação na história da computação. Segundo Muchow (2004), dois anos após a apresentação do Java ao mercado, uma nova edição foi lançada, a Java 2 Enterprise Edition (J2ME), fornecendo suporte para aplicações em nível empresarial de larga escala. Ainda segundo Moraes (2009, p. 3), o Java continuou e continua crescendo e hoje é com certeza um padrão para o mercado oferecendo qualidade, performance e segurança ainda sem nenhum competidor a altura. O Java tornou-se popular pelo seu uso na Internet e hoje possui seu ambiente de execução presente em web browsers, *mainframes*, sistemas operacionais, celulares, *palmtops* e cartões inteligentes, entre outros.

Conforme Fioresi (2007) as plataformas Java atualmente disponíveis podem ser dividas em:

a) Java 2 Standard Edition (J2SE): tecnologia projetada para computadores pessoais e ambientes de trabalho;
b) Java 2 Enterprise Edition (J2EE): tecnologia direcionada para aplicações baseadas no servidor, contendo suporte interno para JSP (JavaServer Pages), XML (eXtensible Markup Language) e *servlets*;
c) Java 2 Micro Edition (J2ME): tecnologia direcionada para dispositivos com poucos recursos computacionais como, por exemplo, palms e telefones celulares.

A Figura 2 apresenta as edições da linguagem Java de acordo com os seus alvos de aplicação.


Fonte: Fioresi (2000)

Figura 2 - Edições da linguagem Java e seus respectivos alvos de aplicação

### 4.2.1 JAVA 2 MICRO EDITION (J2ME)

De acordo com Fonseca (2002), o Java 2 Platform Micro Edition (J2ME) na verdade é um conjunto de especificações que tem por objetivo disponibilizar uma JVM, API e ferramentas para equipamentos portáteis e qualquer dispositivo com poder de processamento menor que os atuais computadores de mesa. A plataforma J2ME oferece para tanto uma máquina virtual Java, chamada de KVM, pequena o bastante para ser suportada dentro das restrições de memória destes dispositivos.

Segundo Fioresi (2007), a arquitetura da plataforma J2ME é dividida em três camadas: Máquina Virtual, Configurações e Perfis, como na Figura 3.


Fonte: Fioresi (2007)

Figura 3 - A Arquitetura Java 2 Micro Edition (J2ME)

O Profile, ou perfil, define um conjunto de bibliotecas específicas para classes de dispositivos. Segundo Corbera (2005), a diferença entre Configuração e Perfil é que a Configuração descreve de forma geral uma família de dispositivos, enquanto o Perfil fica mais específico para um tipo particular de aparelho em uma dada família. O Perfil tão somente acrescenta funcionalidades àquele aparelho.

A configuração é um conjunto de bibliotecas básicas disponíveis para o programador. Ela também define qual o nível de serviços e funcionalidades oferecidos pela máquina virtual.

Uma é a *Connected Limited Device Configuration* (CLDC), que fornece um conjunto de Java API para aplicações sem fio, ou seja, que sejam suportadas pelo dispositivo móvel. Essa especificação fornece as classes responsáveis pela conexão, entrada e saída de dados, classes de manipulações de *strings* e de operações matemáticas.

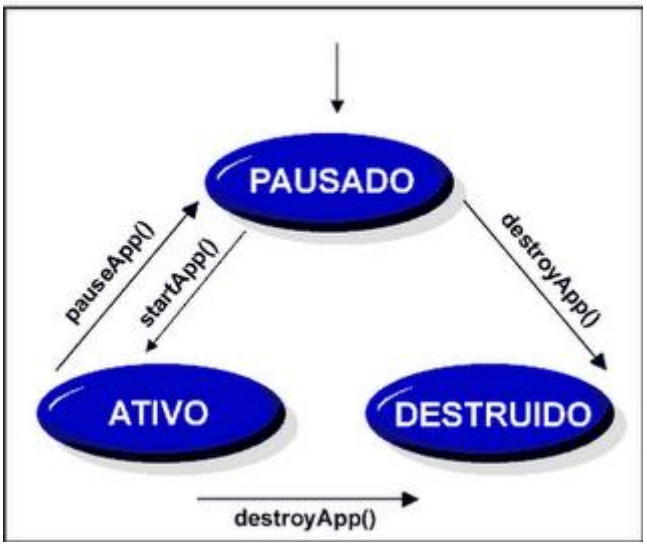
A outra é a *Mobile Information Device Profile* (MIDP), que oferece uma biblioteca de interface gráfica para o dispositivo móvel. Essa especificação provê ainda as classes para memória persistente e algumas classes que definem objetos de formulário.

Além das configurações e perfis, ainda existem as bibliotecas chamadas pacotes opcionais, que são bibliotecas de programação específicas a uma determinada tecnologia. Elas aumentam a capacidade do ambiente, caso estejam implementadas no dispositivo.

### 4.2.2 MIDlets

Segundo Muchow (2004), chama-se MIDlet é um aplicativo Java para executar em um dispositivo móvel. Uma classe MIDlet constrói um aplicativo, onde os métodos dessa classe são acessados pelo gerenciador de aplicações. Quando mais de uma MIDlet encontram-se em uma mesma aplicação, dá-se o nome de MIDlet Suite.

Ainda segundo Muchow (2004), o ciclo de vida de uma MIDlet é dividido em três estados: Pausa, Ativa e Destruída. A MIDlet pode passar por cada um desses estados. A Figura 4 apresenta a máquina de estados que compõem este ciclo de vida.




Fonte: Javamovel (2009)

Figura 4 - Ciclo de vida do MIDlet

Quando a aplicação é iniciada o gerenciador de aplicações passa o estado da aplicação para ativo. Enquanto a aplicação permanecer neste estado, ela poderá ser pausada por um evento externo ao aplicativo ou pelo próprio usuário. Quando a aplicação for encerrada seu estado passa então para destruído.

# 5 DESENVOLVIMENTO

Neste capítulo serão vistas as funcionalidades do protótipo desenvolvido, bem como as ferramentas utilizadas. Serão apresentados os requisitos e a especificação através da *Unified Modeling Language* (UML). O protótipo deve ser utilizado em dispositivos móveis que possuam suporte a conexão de dados e suportem a tecnologia J2ME, com a configuração CLDC e o perfil MIDP.

## 5.1 REQUISITOS PRINCIPAIS DO PROTÓTIPO

Através do protótipo deverá ser possível efetuar a administração de recursos básicos do servidor, bem como a monitoração de recursos do mesmo. Podem-se citar os requisitos principais, que devem estar presentes no protótipo:

a) permitir a inclusão, alteração e exclusão de usuários no servidor Linux em que se está administrando (Requisito Funcional – RF);
b) possibilitar o controle de acesso à internet, à rede *wireless* e à rede interna para os usuários cadastrados (RF);
c) permitir a consulta de estatísticas de utilização da rede, utilização do processador e memória do servidor (RF);
d) permitir a monitoração dos serviços instalados no servidor (servidor web, e-mail, transferência de arquivos, etc.), bem como alertas de segurança (RF);
e) utilizar chave asimétrica para a autenticação do usuário no servidor (RF);
f) utilizar o protocolo SSH versão dois para comunicação segura com o servidor (RF);
g) utilizar as ferramentas de desenvolvimento providas pelo Symbian (Requisito Não Funcional - RNF);
h) ser desenvolvido em linguagem Java (RNF);
i) utilizar o ambiente de desenvolvimento Netbeans (RNF);
j) ser implementado usando a análise orientada a objetos (RNF);
k) possuir interface intuitiva para usuários com pouco ou nenhum conhecimento em gerenciamento de redes (RNF).

## 5.2 ESPECIFICAÇÃO

Para especificação do protótipo foi utilizado a UML, usando os diagramas de casos de uso, de classes e de atividades. Para criação destes diagramas foi utilizado o software Enterprise Architect da Sparx Systems.

### 5.2.1 Diagrama de casos de uso

Na Figura 5 são demonstrados os casos de uso da ferramenta que especificam como o papel do administrador da rede interage com a aplicação para administrar os usuários do servidor.

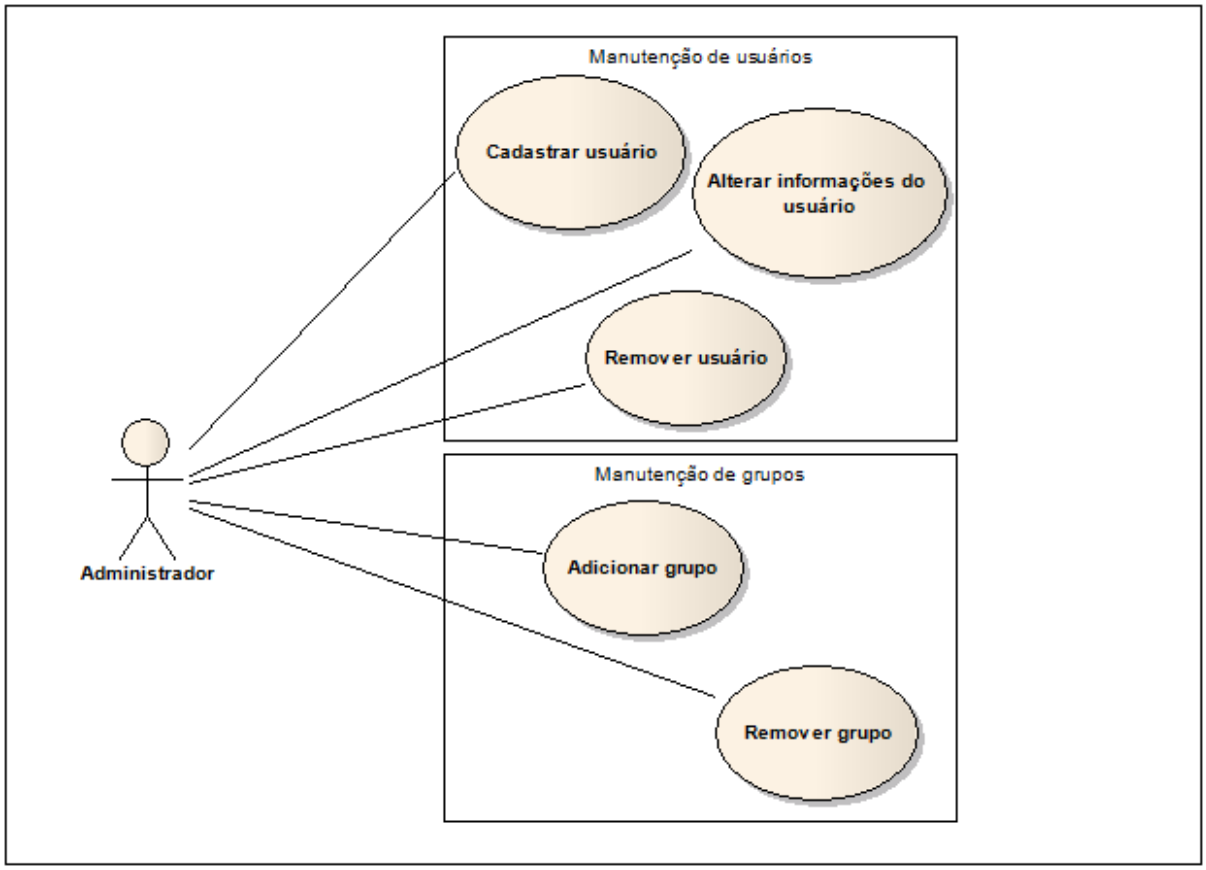



Figura 5 – Diagrama de casos de uso

Cada um destes cinco casos de uso tem a seguinte função:

a) cadastrar usuário: este caso de uso permite ao administrador incluir um novo usuário no servidor;

b) alterar informações do usuário: neste caso de uso o administrador pode alterar

informações como nome e senha, bem como bloquear um usuário;

c) remover usuário: este caso de uso permite ao administrador remover um usuário do servidor;

d) adicionar grupo: este caso de uso permite ao administrador adicionar um novo grupo;

e) remover grupo: este caso de uso permite ao administrador remover um grupo cadastrado.

Na Figura 6 são demonstrados os casos de uso da ferramenta que especificam como o papel do administrador da rede efetua a monitoração dos recursos, bem como a ferramenta gerencia os alertas para o administrador.

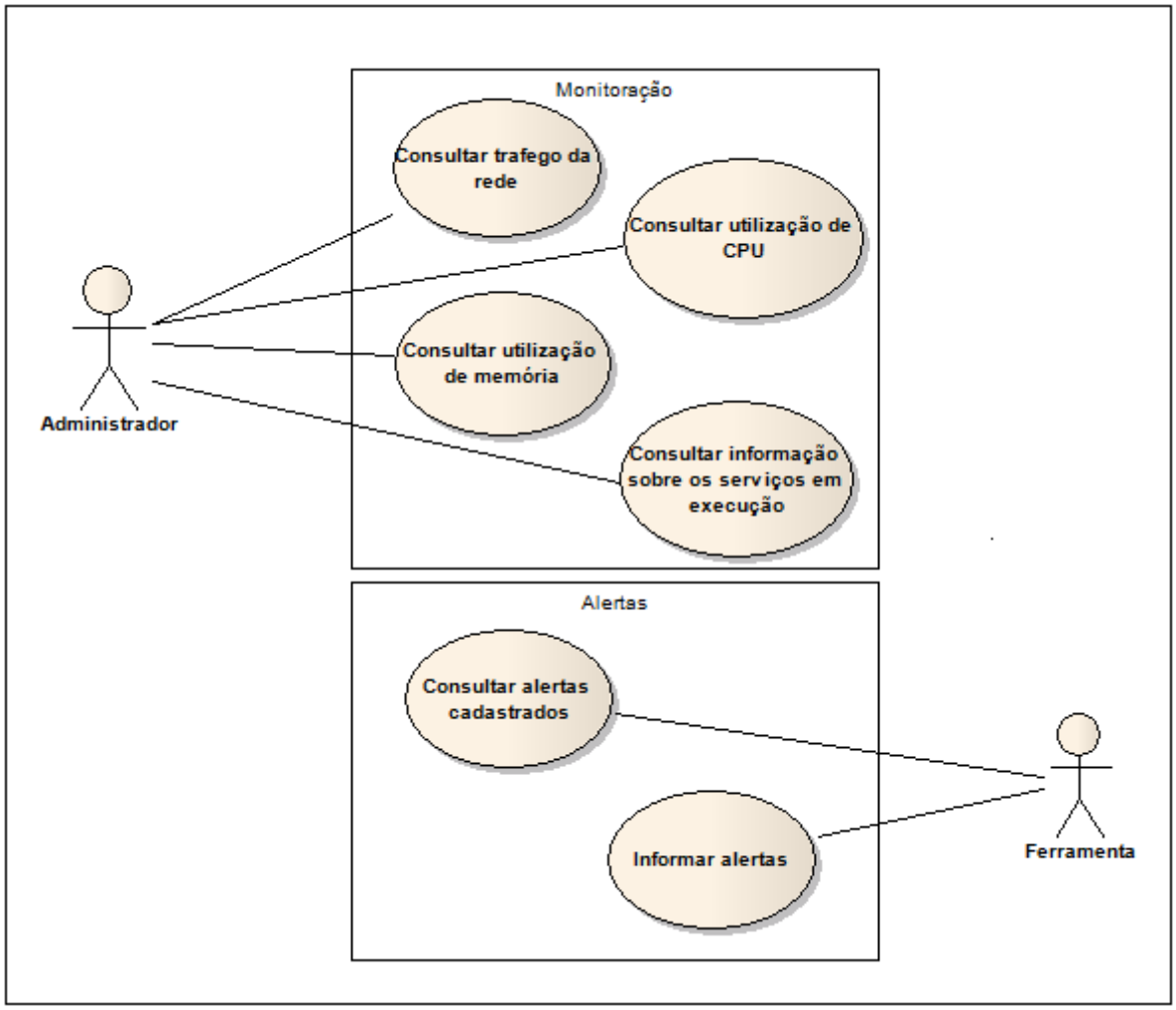



Figura 6 – Diagrama de casos de uso

Cada um destes seis casos de uso tem a seguinte função:

a) consultar tráfego da rede: neste caso de uso o administrador pode verificar a taxa de transmissão de dados no servidor;

b) consultar utilização de CPU: neste caso de uso o administrador pode verificar a

utilização do processador no servidor;

c) consultar utilização de memória: este caso de uso permite ao administrador consultar a utilização de memória do servidor;
d) consultar informação dos serviços em execução: neste caso de uso o administrador pode consultar o status dos serviços em execução no servidor;
e) consultar alertas cadastrados: neste caso de uso a ferramenta consulta os alertas configurados no protótipo, consulta as informações necessárias no servidor que está sendo administrado e atualiza as informações no celular;
f) informar alertas: neste caso de uso a ferramenta, com base nas informações coletadas no servidor, informa ao administrador sobre os alertas ocorridos.

### 5.2.2 Diagrama de classes

O diagrama de classes descreve como serão divididas as mesmas na implementação, apresentando os principais atributos e métodos. A Figura 7 apresenta as classes principais para o gerenciamento do servidor.

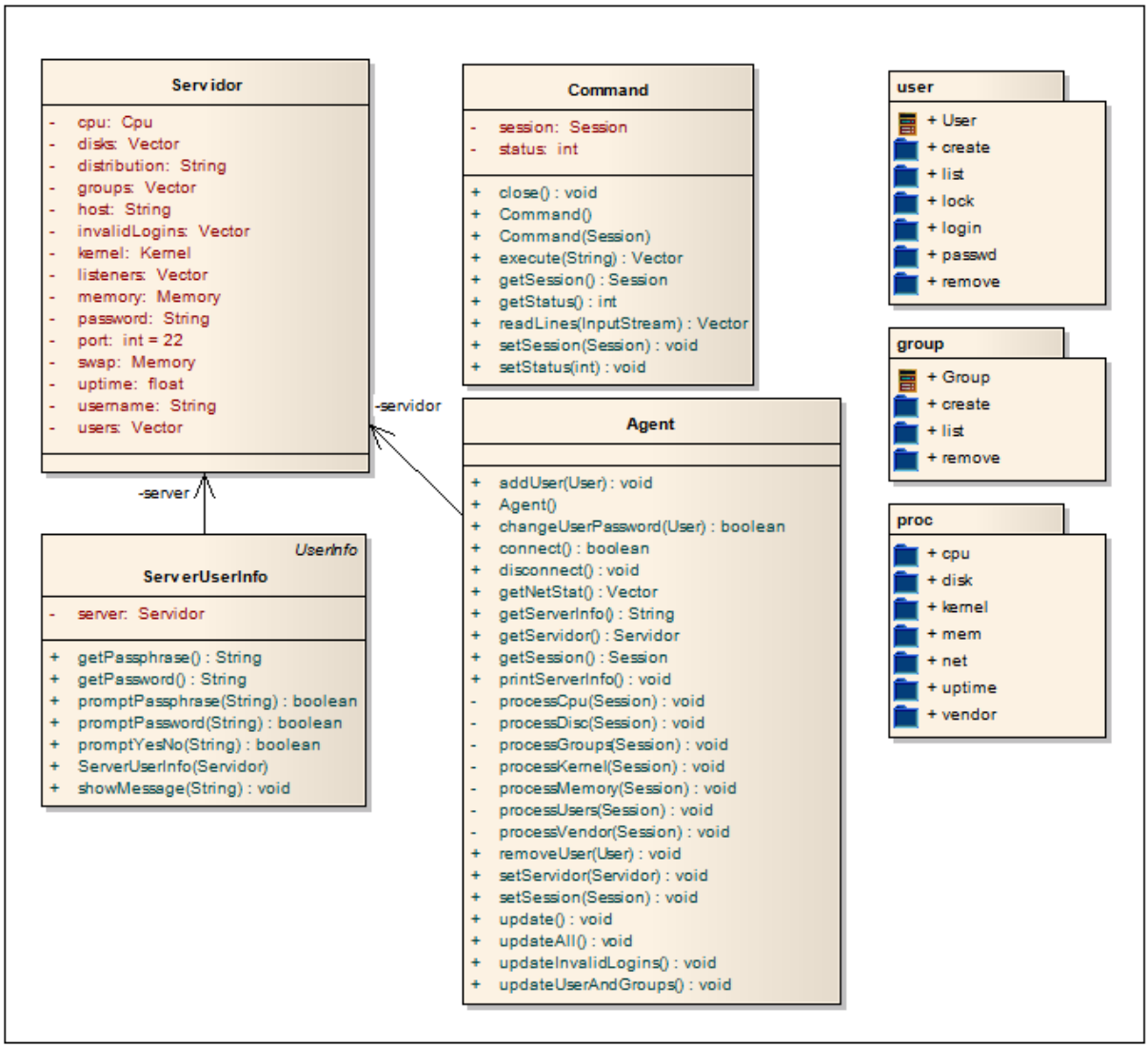



Figura 7 - Classes para o gerenciamento do servidor

As classes apresentadas possuem as seguintes funções:

a) a classe `Servidor` armazena os dados coletados do servidor como memória, disco e processamento;

b) a classe `Command` é utilizada para padronizar a realização de comandos no servidor;

c) a classe `ServerUserInfo` armazena as informações de *login* do administrador para efetuar a conexão com o servidor. Esta classe é requerida pela biblioteca JSch;

d) a classe `Agente` é responsável pela chamada e controle das consultas no servidor.

A Figura 8 apresenta as classes utilizadas para o gerenciamento dos usuários do sistema. Através destas classes o sistema permite ao administrador a inclusão, atualização e exclusão de usuários.

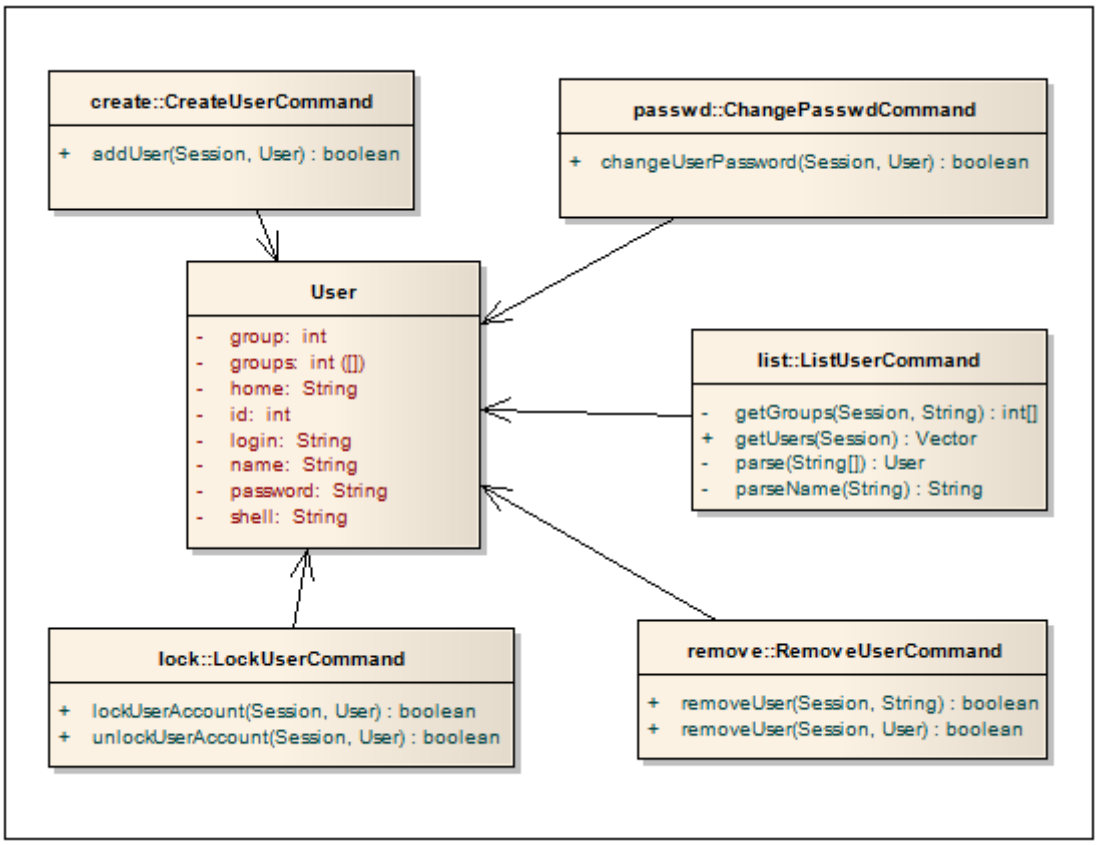



Figura 8 - Classes para o gerenciamento de usuários

As classes apresentadas possuem as seguintes funções:

a) a classe `CreateUserCommand` é responsável pela execução do comando de criação de usuários no servidor;

b) a classe `ChangePasswdCommand` é responsável pela execução do comando de alteração de senha dos usuários;

c) a classe `ListUserCommand` é responsável pela consulta dos usuários cadastrados no servidor;

d) a classe `RemoveUserCommand` é responsável pela execução dos comandos de exclusão de usuários;

e) a classe `LockUserCommand` é responsável pela execução do comando de bloqueio dos usuários do servidor;

f) a classe `User` armazena as informações sobre os usuários consultados no servidor.

A Figura 9 apresenta as classes responsáveis pelo gerenciamento dos grupos de usuários.

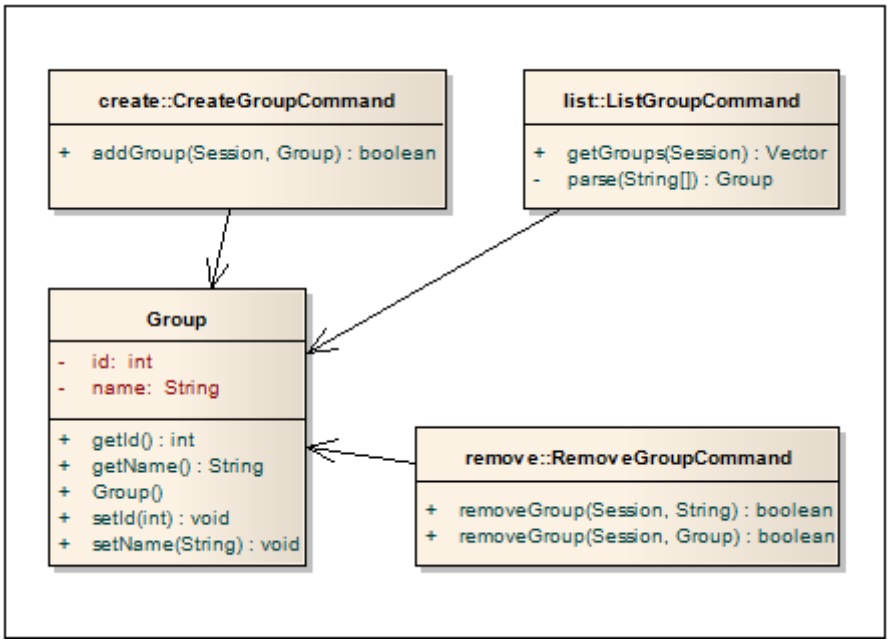



Figura 9 - Classes para o gerenciamento de grupos

As classes apresentadas possuem as seguintes funções:

a) a classe `CreateGroupCommand` é responsável pela execução do comando de criação de grupos no servidor;

b) a classe `ListGroupCommand` é responsável pela consulta dos grupos cadastrados no servidor;

c) a classe `RemoveGroupCommand` é responsável pela execução dos comandos de exclusão de grupos;

d) a classe `Group` armazena as informações sobre os grupos cadastrados no servidor.

A Figura 10 apresenta as classes para o gerenciamento de alertas que avisam ao administrador determinadas condições do sistema que necessitam uma atenção especial.

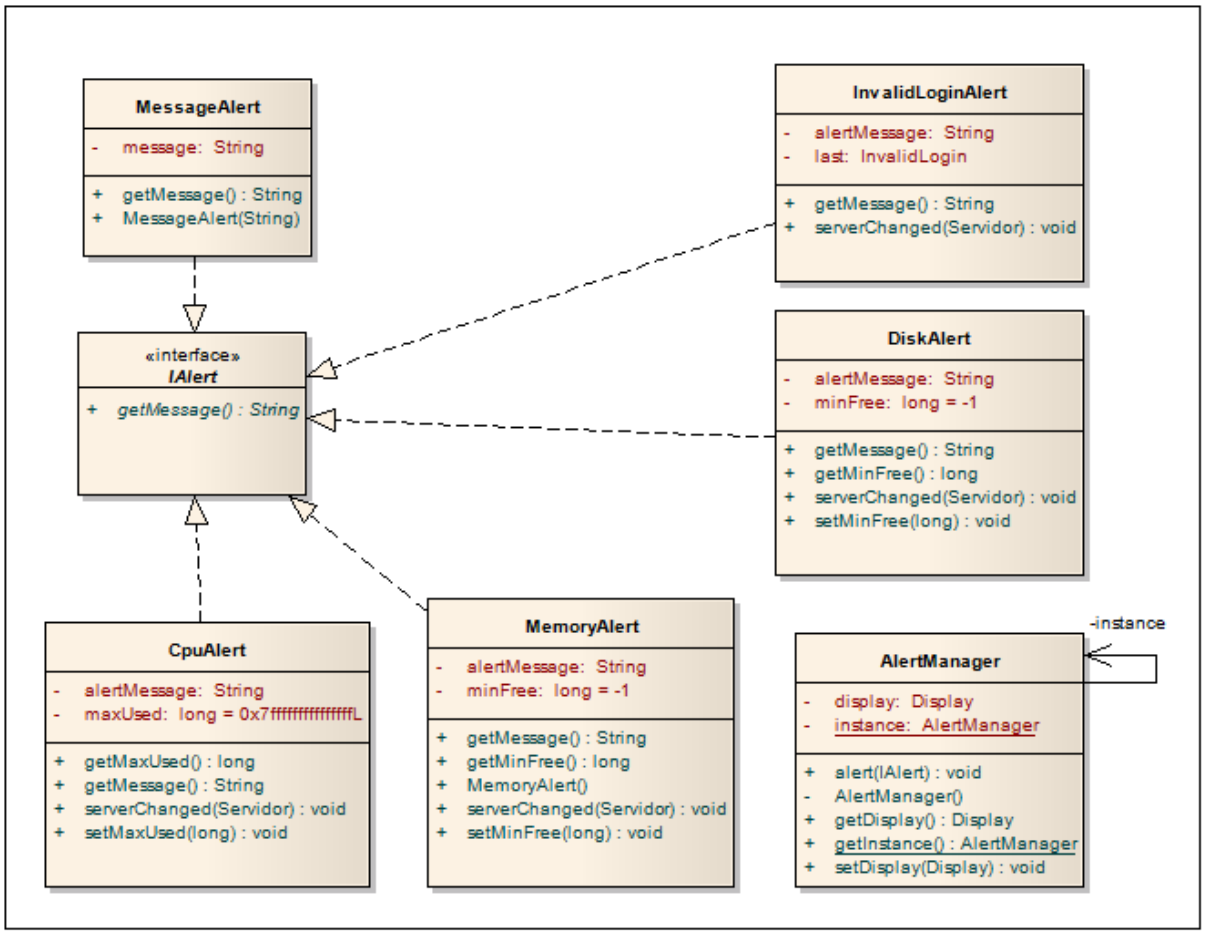



Figura 10 - Classes para o gerenciamento de alertas

As classes apresentadas possuem as seguintes funções:

a) a classe `MessageAlert` é responsável por apresentar avisos ou erros no sistema;

b) a interface `IAlert` é utilizada para padronizar as classes de alerta;

c) a classe `InvalidLoginAlert` é responsável por enviar alertas sobre tentativas de *login* inválido do servidor;

d) a classe `DiskAlert` é responsável por enviar alertas sobre o nível de utilização do disco no servidor;

e) a classe `AlertManager` é responsável por gerenciar todos os alertas do protótipo;

f) a classe `MemoryAlert` é responsável por enviar alertas sobre o nível de utilização da memória do servidor;

g) a classe `CpuAlert` é responsável por enviar alertas sobre o nível de utilização do processamento no servidor.

A Figura 11 apresenta as classes responsáveis por consultar as informações sobre versão do *kernel*, distribuição, utilização do processador, memória e rede.

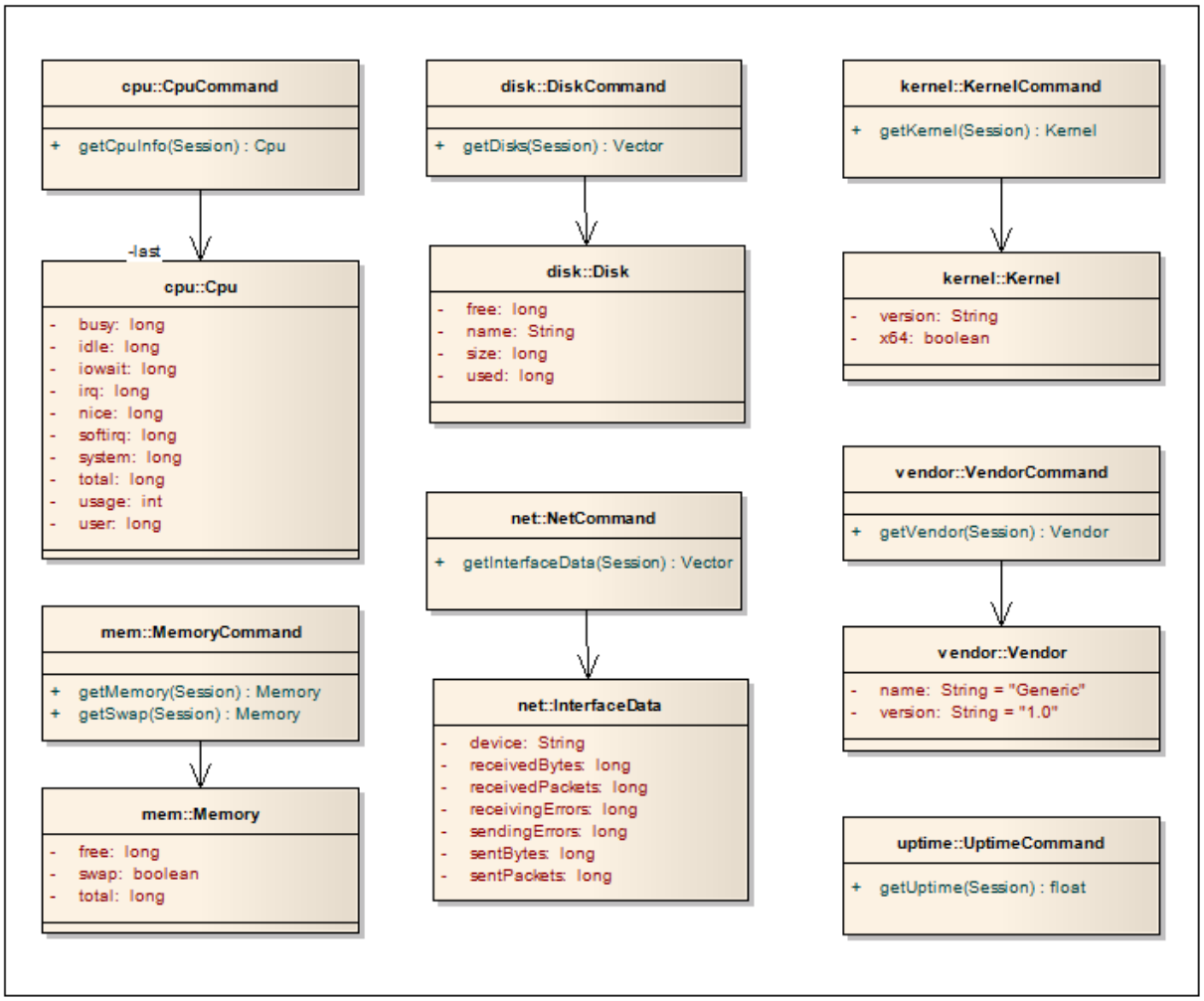



Figura 11 – Classes para consulta de informações do sistema

As classes apresentadas possuem as seguintes funções:

a) a classe `CpuCommand` é responsável pela consulta das informações sobre o processamento no servidor;

b) a classe `DiskCommand` é responsável pela consulta das informações sobre o disco rígido do servidor;

c) a classe `KernelCommand` é responsável pela consulta da versão do *kernel* utilizado servidor;

d) a classe `MemoryCommand` é responsável pela consulta das informações sobre a memória do servidor;

e) a classe `NetCommand` é responsável pela consulta das informações sobre a utilização da rede no servidor;

f) a classe `VendorCommand` é responsável pela consulta da versão do Linux instalado no servidor;

g) a classe `UptimeCommand` é responsável pela consulta do tempo que o servidor

permanece ligado;

h) a classe `Cpu` é responsável pela por armazenar as informações sobre o processamento;
i) a classe `Disk` é responsável pela por armazenar as informações sobre o disco;
j) a classe `Memory` é responsável pela por armazenar as informações sobre a memória;
k) a classe `InterfaceData` é responsável pela por armazenar as informações sobre a utilização da interface de rede;
l) a classe `Vendor` é responsável pela por armazenar o nome da distribuição instalada no servidor;
m) a classe `Kernel` é responsável pela por armazenar a versão do *kernel* utilizada no servidor.

## 5.3 IMPLEMENTAÇÃO

Esta seção contém o detalhamento sobre a implementação do protótipo. O tópico inicial identifica as ferramentas utilizadas. O tópico seguinte apresenta a operacionalidade do protótipo.

### 5.3.1 Ferramentas utilizadas na implementação

Para a implementação do protótipo foi utilizada a linguagem J2ME, com a configuração CLDC 1.1 e o perfil MIDP 2.0, e o ambiente de desenvolvimento Netbeans 6.8 com o *plugin* para desenvolvimento de aplicações móveis. Pulsar. Na Figura 12 pode ser visto a tela principal do Netbeans, apresentando o fluxo das telas. O desenvolvimento da interface gráfica com o usuário foi realizado com a biblioteca J2ME Polish.

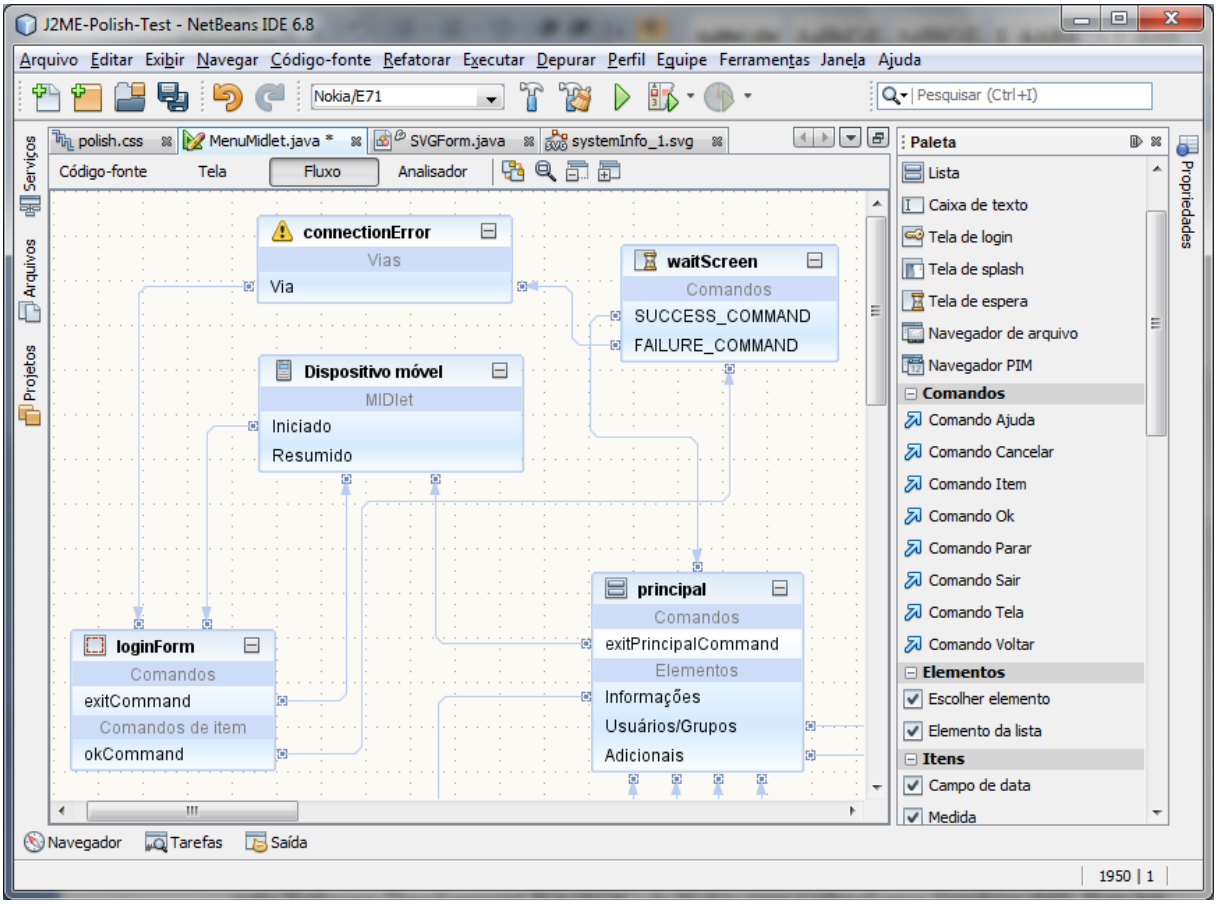



Figura 12 - Netbeans

Os testes fora do dispositivo móvel foram realizados no ambiente de emulação provido pelo *Software Development Kit* (SDK) da Nokia para a plataforma Symbian S60. Este kit provê um conjunto de ferramentas para desenvolvimento, depuração e testes do software. O emulador presente no SDK, que pode ser visto na Figura 13, permite o teste da aplicação com acesso à recursos similares a um aparelho móvel contendo o sistema operacional Symbian, como por exemplo o acesso à rede simulando um acesso *wireless*.


Figura 13 - Emulador Nokia SDK

### 5.3.2 Bibliotecas auxiliares

A seguir são apresentadas as bibliotecas auxiliares Jsch, Javolution, *Scalable 2D Vector Graphics* API e J2ME POLISH que foram utilizadas no desenvolvimento deste protótipo.

Jsch é uma biblioteca que implementa o protocolo de comunicação *Secure SHell* (SSH) versão dois em linguagem Java. O principal objetivo da biblioteca é prover uma API para conexões que utilizam o protocolo SSH versão dois. Jsch é uma biblioteca livre, distribuída com a licença Berkeley Software Distribution (BSD), podendo ser utilizada em aplicações públicas e comerciais (JCRAFT, 2005).

Segundo Ferbers (2008) Jsch é uma das únicas implementações, com o código fonte livre, puramente desenvolvido em Java, para a especificação de SSH. Devido a lançamentos e correções frequentes, curto espaço de tempo para os questionamentos aos desenvolvedores, a

biblioteca tornou-se mais e mais estável.

Ainda segundo Ferbers (2008), a JSch fornece uma biblioteca de baixo nível para o protocolo SSH. Essencial ao programador que precisa ter total controle sobre a conexão SSH. No entanto, um código extenso é necessário, mesmo para tarefas triviais, como executar um comando simples no *host* remoto. O programador tem que cuidar de todos os detalhes mínimos.

Javolution é uma biblioteca para desenvolvimento de sistemas em tempo-real e embarcados. Ela provê um conjunto de classes de alto desempenho e com comportamento altamente tempo-determinístico, para conformidade com ambiente tempo-real, coleções e estruturas fundamentais, manipulação de texto, entrada e saída de dados e XML. Javolution é um *software* de código aberto distribuído sob licença BSD (D'ÁVILA, 2007).

*Scalable 2D Vector Graphics* API, ou JSR-226 como é conhecida, é uma *API* para aplicações *Scalable Vectorial Graphics* (SVG) em dispositivos compatíveis com J2ME. A JSR 226 teve sua versão final aprovada no ano de 2006, segundo a página oficial da JSR, e implementa especificamente o perfil SVG Tiny.

O J2ME Polish é um conjunto de ferramentas que visa auxiliar o programador a lidar com vários aspectos da interface gráfica na criação de aplicações para dispositivos de pequeno porte (FERNANDES, 2007). O J2ME Polish possui excelentes recursos em termos de interface gráfica bem como para a persistência de dados e comunicação em dispositivos móveis se comparado com as classes disponíveis na API J2ME disponibilizada pela Sun Microsystems (BIRK, 2007, p. 25).

Criado por volta de 2004 por Robert Virkus, o J2ME Polish vem para suprir o problema de compatibilidade com os diversos dispositivos existentes no desenvolvimento de softwares para dispositivos móveis.

O projeto J2ME Polish pode ser dividido em cinco módulos principais descritos a seguir:

a) `Lush`: biblioteca de componentes gráficos para o desenvolvimento de interfaces gráficas;
b) `Janus`: conjunto de ferramentas para o desenvolvimento de aplicações para diferentes plataformas de dispositivos;
c) `Touch`: componentes para comunicação e acesso remoto;
d) `Trunk`: componentes para persistência de dados;
e) `Marjory`: base de dados de dispositivos móveis.

### 5.3.3 Principais funções do protótipo

Nesta seção são apresentadas as principais rotinas do protótipo. No Quadro 5 – Código para conexão SSH com o servidor é apresentada a rotina de conexão SSH com o servidor. Em primeiro lugar é feita a inicialização da biblioteca JSch, e em seguida um teste se existe uma conexão ativa. No caso de não existir restrições, uma nova conexão é inicializada.

```
/**
 * Cria uma nova conexao SSH com o servidor
 * @return Session
 */
public boolean connect() throws JSchException {
	// instanciar biblioteca de conexao
	if (jsch == null) {
		jsch = new JSch();
	}
	// verificar se existe conexao ja realizada com o mesmo usuario
	String foo = user + "@" + host;
	if (last_uh != null && last_uh.equals(foo) && session != ull) {
		return true;
	}
	// se existir conexao antiga, finalizar
	if (session != null) {
		try {
			session.disconnect();
		} catch (Exception e) {
			e.printStackTrace();
		}
		session = null;
	}
	// criar nova sessao
	Session _session = jsch.getSession(user, host, port);
	_session.setSocketFactory(new J2MESocketFactory());
	MyUserInfo uInfo = new MyUserInfo(passwd);
	_session.setUserInfo(uInfo);
	_session.connect();
	last_uh = foo;
	session = _session;
	// retorna true se conectado
	return isConnected();
}
```

Quadro 5 – Código para conexão SSH com o servidor

No Quadro 6 é apresentada a rotina para execução de comandos no servidor através do protocolo SSH. Primeiramente um canal de comunicação é criado, através de uma sessão (conexão) já existente. Após a execução, o retorno do comando é passado para uma rotina quebrar o resultado em linhas, facilitando a análise do conteúdo.

```java
/**
 * Executa um comando na sessao aberta e retorna o resultado
 *
 * @param command
 * @return Vector resultado
 * @throws IOException
 * @throws JSchException
 */
public Vector execute(String command) throws IOException, JSchException
{

	// abrir um canal de execucao de comandos
	Channel channel = getSession().openChannel("exec");
	channel.setInputStream(null);
	((ChannelExec) channel).setErrStream(System.err);
	// informar comando
	((ChannelExec) channel).setCommand(command);

	// retornar resposta do servidor
	InputStream in = channel.getInputStream();
	channel.connect();

	// capturar linhas da resposta
	Vector result = readLines(in);
	channel.disconnect();

	// retornar linhas
	return result;
}
```

Quadro 6 – Rotina para execução de comandos

O Quadro 7 apresenta a rotina responsável por analisar o retorno do comando e quebrar o conteúdo em um vetor de linhas. Cada linha apresentada no vetor representa uma linha de informação recuperada do retorno do comando executado no servidor. O fim de cada linha do resultado é demarcado pelos caracteres de controle Carriage Return (CR) e Line Feed (LF).

```
/**
 * ler inputstream separar em linhas
 * @param input
 * @return Vector
 */
public Vector readLines(InputStream input) {
	Vector dict = new Vector();
	try {
		InputStreamReader isr =
			new InputStreamReader(input, "UTF-8");
		StringBuffer buffer = new StringBuffer();
		int ch;
		while ((ch = isr.read()) != -1) {
			char th = (char) ch;
			if (th == '\n') {
				dict.addElement(buffer.toString());
				buffer.delete(0, buffer.length());
			}// end if
			else if (th != '\r') {
				buffer.append(th);
			}
		}// end while
		isr.close();
	}
	catch (UnsupportedEncodingException e) {
		System.err.println(e);
	}
	catch (IOException e) {
		e.printStackTrace();
	}
	return dict;
}
```

Quadro 7 – Rotina para separar o resultado em linhas

No Quadro 8 é apresentado um exemplo de comando para retornar o tempo total que o servidor está *online*.

```
/**
 * Retornar tempo total que o servidor esta online
 * @param session
 * @return tempo
 * @throws JSchException
 * @throws IOException
 */
public float getUptime(Session session) throws JSchException,
	IOException {

	Command cmd = new Command(session);
	Vector lines = cmd.execute("cat /proc/uptime");
		if( lines != null && lines.size() > 0 ) {
		String line = String.valueOf(lines.elementAt(0));
		int pos = line.indexOf(' ');
		if(pos != -1) {
			String part = line.substring(0, pos).trim();
			return Float.parseFloat(part);
		}
	}
	return 0;
}
```

Quadro 8 – Comando para retornar o tempo que o servidor está ativo

O comando `cat /proc/uptime` é executado no servidor retornando a quantidade total em segundos que o servidor ligado. As linhas retornadas são analisadas e o resultado que está em formato *string* então é convertido para o formato numérico e retornado como resultado da função.

### 5.3.4 Operacionalidade do protótipo

Nesta seção é apresentado o funcionamento do protótipo. O protótipo é composto pelas seguintes telas:

a) tela de *login*;
b) informações do servidor;
c) gerenciamento de usuários e grupos;
d) gerenciamento de conexão;
e) gerenciamento de alertas.

Na Figura 14 é demonstrada a tela de *login* do sistema. Nesta tela o administrador pode escolher na lista o servidor no qual deseja efetuar a conexão.

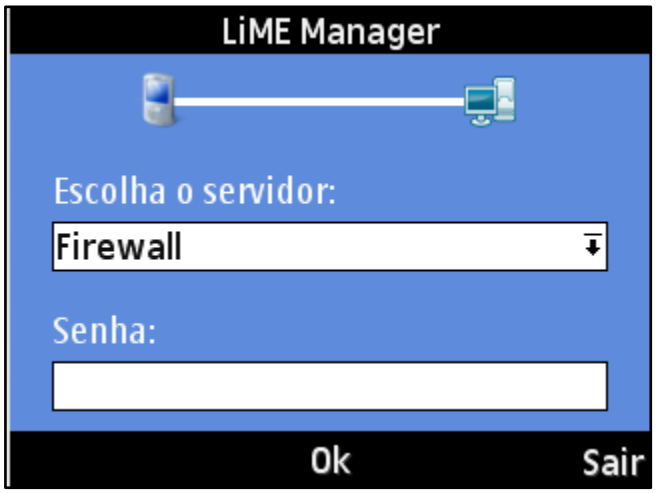



Figura 14 - Tela de *login*

Após a escolha do servidor, demonstrado na Figura 15, e o preenchimento da senha, é apresentada a tela inicial do sistema.

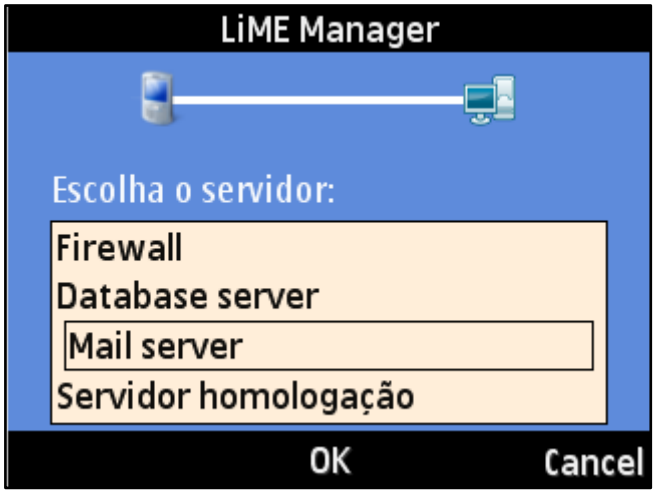



Figura 15 - Escolha do servidor

Na Figura 16 é apresentado tela inicial do protótipo contendo o menu de opções existentes.


Figura 16 - Tela inicial

Na Figura 17 é apresentada a tela com as informações sobre a utilização de memória, processamento, utilização do disco, versão do *kernel* e distribuição utilizada.

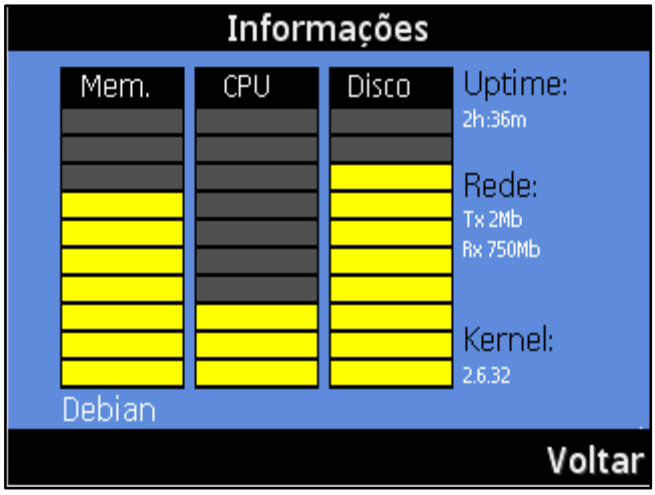



Figura 17 - Informações do servidor

A Figura 18 apresenta as opções de administração dos usuários e grupos do sistema. As opções disponíveis no menu são:

a) adicionar usuário: fornece ao administrador a possibilidade de adicionar novos usuários ao servidor;
b) excluir usuário: fornece ao administrador a funcionalidade de excluir usuários existentes no sistema;
c) alterar usuário: permite a alteração de dados e o bloqueio de usuários do sistema;
d) criar grupo: permite a criação de grupos de usuários;
e) remover grupo: permite a exclusão de grupos de usuários.


Figura 18 - Administração de usuários e grupos

Na Figura 19 é apresentada a tela para inclusão de novos usuários. Para incluir um novo usuário o administrador deve informar o nome do usuário, senha, grupo de acesso e IP da máquina do usuário.


Figura 19 - Tela para inclusão de usuário

Na Figura 20 é apresentada a tela de serviços adicionais do sistema. Esta tela possui as seguintes opções:

a) serviços: permite ao administrador visualizar os serviços em execução;
b) acesso a internet: permite ao administrador gerenciar o acesso a internet pelos usuários;
c) gerenciamento de alertas: permite ao administrador ajustar os parâmetros dos alertas do sistema.

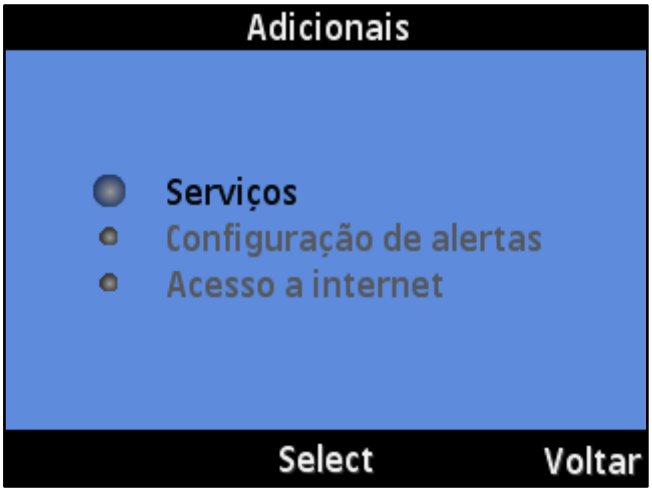



Figura 20 - Serviços adicionais

A Figura 21 apresenta a tela para configuração dos alertas do sistema. Esta tela permite ao administrador a configuração do percentual de espaço em disco livre necessário para que o sistema efetue um alerta, o percentual máximo de utilização do processamento e o tempo entre cada consulta ao servidor.

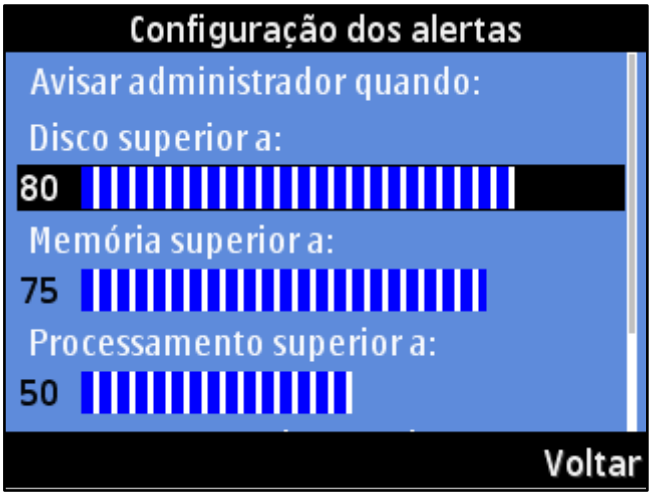



Figura 21 - Tela de configuração dos parâmetros

A Figura 22 - Mensagens de alerta demonstra uma mensagem de erro de conexão apresentada durante o *login* do administrador caso o servidor escolhido esteja indisponível.

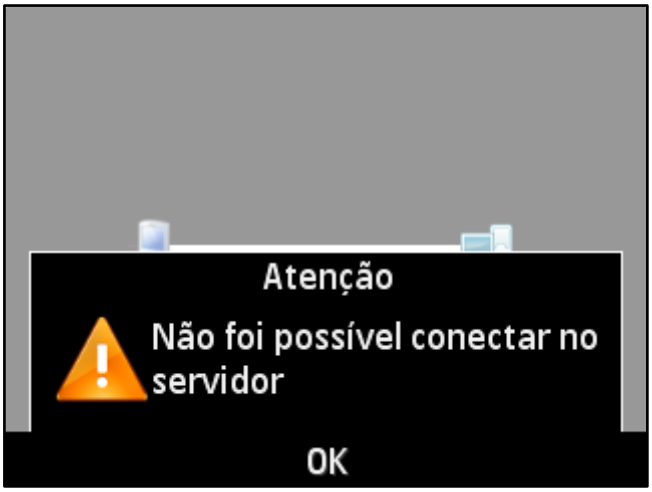



Figura 22 - Mensagens de alerta de conexão

A Figura 23 apresenta a mensagem de alerta para falta de espaço em disco no servidor Outros tipos de mensagens apresentadas são a de falta de espaço em disco, tentativas inválidas de *login* e alta utilização de processamento no servidor.

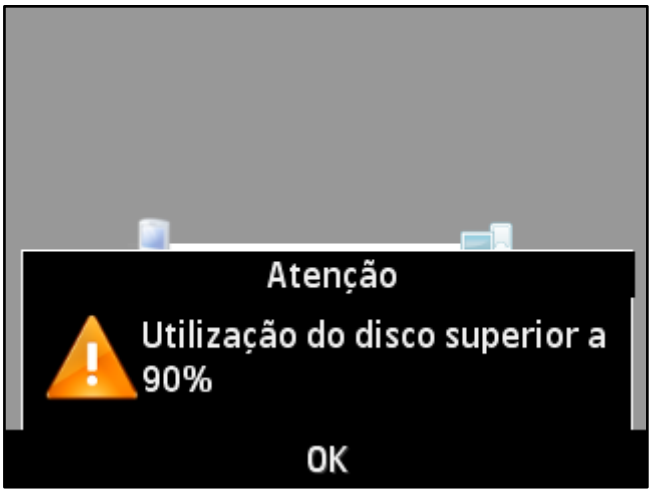



Figura 23 - Mensagens de alerta disco

## 5.4 RESULTADOS E DISCUSSÃO

O presente trabalho é uma demonstração de como a tecnologia móvel pode auxiliar nas tarefas do dia-a-dia. Os dispositivos móveis aliados à tecnologia *wireless* possibilitam uma grande mobilidade nas tarefas que antes eram restritas a estações fixas. O protótipo desenvolvido utilizou a tecnologia J2ME no desenvolvimento para possibilitar sua utilização na grande diversidade de dispositivos existentes no mercado. Também foram utilizadas as

ferramentas de desenvolvimento disponibilizadas pela Nokia para a arquitetura do Symbian S60 que facilitaram os testes e depuração de problemas.

Nas pequenas empresas de Tecnologia da Informação (TI) geralmente não existe uma pessoa específica para efetuar o gerenciamento da rede corporativa. Neste caso o funcionário que executa esta função não está disponível em tempo integral para analisar possíveis problemas ou efetuar manutenções solicitadas. Por este motivo foi escolhido o desenvolvimento deste protótipo, para facilitar nas tarefas básicas da administração de redes.

O protótipo foi submetido a testes com distribuições Linux em máquinas virtuais e no *firewall* corporativo de uma empresa de TI. O protótipo permite o gerenciamento dos servidores através de uma rede *wireless* interna, ou mesmo através de uma conexão 3G.

Na Figura 24 observa-se as informações de um servidor Linux utilizado como firewall e *proxy* de internet. Pode-se verificar o tempo em que o servidor está em execução, bem como a utilização dos recursos como processamento, disco e memória.

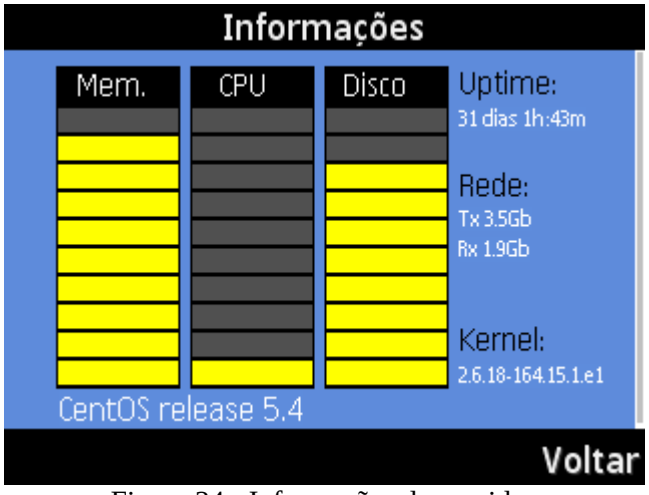



Figura 24 - Informações do servidor

Na Figura 25 é demonstrada a inclusão de um novo usuário no servidor. Após sua inclusão, seu *login* pode ser visto na lista de usuários apresentado na Figura 26.


Figura 25 - Adicionar usuário

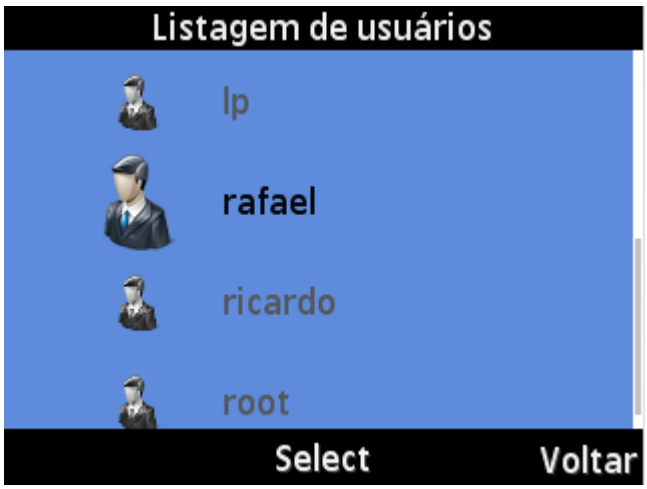



Figura 26 - Lista de usuários

O teste apresentado nesta seção, valida a implementação do protótipo especificado. Sendo assim, os objetivos específicos do presente trabalho foram atingidos.

# 6 CONCLUSÕES

No presente trabalho foi possível a realização de um estudo sobre a tecnologia J2ME aplicado ao gerenciamento de redes em servidores Linux. Foi possível também levantar os principais benefícios e dificuldades no desenvolvimento de aplicações para os dispositivos móveis. O desenvolvimento apresentou dificuldades comuns aos desenvolvedores de aplicações J2ME, como a falta de compatibilidade entre as máquinas virtuais Java para os dispositivos móveis, a limitação de recursos, e a incompatibilidade no desenvolvimento de interface com o usuário nos diversos modelos e tamanhos de celulares e *handhelds*.

Como dificuldade no desenvolvimento do protótipo, foi constatada a falta de classes comuns no desenvolvimento de aplicações *desktop*, como por exemplo, listas e funções comuns para o tratamento de *strings*. Outra dificuldade foram os problemas de inconsistência na utilização dos emuladores de dispositivos móveis.

O protótipo desenvolvido comprovou través de testes, ser um software capaz de auxiliar o gerenciamento de tarefas básicas na administração de um servidor Linux, bem como alertar o administrador sobre problemas com o servidor. Através da implementação do gerenciamento de usuários o administrador pode incluir, excluir, alterar e bloquear usuários do servidor administrado. Através da tela de informações do sistema bem com pelos alertas, o administrador pode monitorar os recursos do sistema operacional. E com o recurso de administração de acesso a rede o administrador consegue facilmente liberar, ou até mesmo bloquear, o acesso dos usuários.

As ferramentas de desenvolvimento na área de dispositivos móveis com suporte a J2ME mostraram-se bem avançadas e adequadas. As ferramentas de apoio disponibilizadas pela Nokia para desenvolvimento de aplicações para Symbian facilitam o trabalho de desenvolvimento e depuração do código, contando com depurador de código, ambiente de emulação, exemplos de código de acesso aos recursos do dispositivo e uma grande quantidade de documentação. Atualmente existe uma grande variedade de documentação e exemplos disponíveis na internet para consulta. O desenvolvimento da interface com o usuário possui vários *frameworks* que visam facilitar o desenvolvimento e minimizar a falta de compatibilidade entre os fabricantes.

Apesar de o trabalho ter sido desenvolvido com as ferramentas de desenvolvimento da plataforma Symbian, visando à utilização em aparelhos dotados deste sistema operacional, a utilização da linguagem Java permite a utilização em outros aparelhos com diferentes

sistemas operacionais. Porém deve ser levado em conta que não foram executados testes em outros aparelhos, e o correto funcionamento não pode ser garantido.

A realização deste trabalho trouxe um bom aprendizado a área de tecnologias móveis e também na coleta de informações existentes nos servidores Linux. Na área das distribuições Linux foi possível verificar as dificuldades de se manter uma rede com diferentes distribuições, e como a LSB contribui para a padronização das mesmas.

## 6.1 EXTENSÕES

Sugere-se como extensão deste trabalho a implementação das seguintes funcionalidades:

a) utilização em dispositivos dotados de interface *touchscreen*;
b) módulo para descoberta de computadores na rede através de protocolo broadcast;
c) cadastro de servidores e geração de chave pública/privada no próprio dispositivo móvel;
d) enviar relatório de alarmes por *e-mail*;
e) contatar usuários por *e-mail*.

# REFERÊNCIAS BIBLIOGRÁFICAS


AZAMBUJA, Wagner. **Symbian**. Ponta Grossa, [2007]. Disponível em: <http://www.vivasemfio.com/blog/symbian/>. Acesso em: 10 maio 2010.

BALL, Bill; DUFF, Hoyt. **Dominando Linux: Red Hat e Fedora**. Rio de Janeiro: Makron Books, 2004. xxxvi, 698 p.

BIRK, João F. G. **Gerência de vendedores móveis na web via redes GSM/GPRS**. 2007. 59 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) - Centro de Ciências Exatas e Naturais, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre. Disponível em: <http://www.inf.pucrs.br/~eduardob/disciplinas/tc/JoaoBirk/TC2/texto.doc>. Acesso em: 13 abr. 2010.

CAMPOS, Augusto. **Profissionais experientes afirmam que todos ganham com o padrão LSB**. Curitiba, [2002]. Disponível em: <http://www.lenep.uenf.br/~bueno/DisciplinaSL/MANUAIS/00-SOFTWARE_LIVRE/REVISTAS/RDL/026/sistema.html/>. Acesso em: 20 abr. 2010.

CARMONA, Tadeu; HEXSEL, Roberto A. **Universidade redes**: torne-se um especialista em redes de computador. São Paulo: Digerati Books, 2005.

CHESWICK, William R.; BELLOVIN, Steven M.; RUBIN, Aviel D. **Firewalls e segurança na internet**: repelindo o hacker ardiloso.2. ed. Porto Alegre : Bookman, 2005. 400 p, il. (Ciência da computação. Redes).

CORBERA, Rodrigo G. **Tutorial de programação J2ME**. [S.l.], [2005]. Disponível em: <http://www.wirelessbrasil.org/wirelessbr/colaboradores/corbera_martins/j2me_01.html>. Acesso em: 11 maio 2010.

COVERT, Adrian; BURGOS, Pedro. **Giz explica**:o que diferencia as seis plataformas de smartphone. São Paulo, [2009]. Disponível em: <http://www.gizmodo.com.br/conteudo/giz-explica-o-que-diferencia-seis-plataformas-de-smartphone-com-symbian>. Acesso em: 17 maio 2010.

D'ÁVILA, Márcio. **Mais software Java para fechar a semana**. Belo Horizonte, [2007]. Disponível em: <http://blog.mhavila.com.br/2007/09/22/mais-software-java-para-fechar-a-semana/>. Acesso em: 13 maio 2010.

DELALANDE, Christophe. **Symbian**. Itajuba, [2003]. Disponível em: <http://www.wirelessbrasil.org/wirelessbr/colaboradores/christophe/symbian.html>. Acesso em: 05 maio 2010.

FERBERS, Daniel. **About JSch open source project**. Campinas, [2008]. Disponível em: <http://techtavern.wordpress.com/2008/09/30/about-jsch-open-source-project/>. Acesso em: 08 maio 2010.

FERNANDES, Jorge F. L. **J2ME Polish**: desenvolvendo interfaces gráficas para aplicações JavaME. Jaguaribe, [2007]. Disponível em: <http://www.devmedia.com.br/articles/viewcomp.asp?comp=8057>. Acesso em: 13 abr. 2010.

FIORESI, Cristiano. **Conceitos básicos das plataformas Java e J2ME**. Vila Velha, [2007]. Disponível em: <http://www.devmedia.com.br/articles/viewcomp.asp?comp=6484>. Acesso em: 05 maio 2010.

FONSECA, Jorge C. **Portando a KVM**. 2002. 64 f. Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) – Centro de Informática, Universidade Federal de Pernambuco, Recife. Disponível em: <http://www.cin.ufpe.br/~tg/2002-1/jcbf.doc>. Acesso em: 16 maio 2010.

IIDA, Renato F. **Desenvolvimento Symbian na plataforma serie 60**. 2006. 93 f. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Engenharia Elétrica, Universidade de Brasília, Brasília. Disponível em: <http://bdtd.bce.unb.br/tedesimplificado/tde_busca/arquivo.php?codArquivo=898>. Acesso em: 05 maio. 2010.

JAVAMOVEL. **Ciclo de vida do MIDlet**.Vila Velha, [2009]. Disponível em: <http://www.javamovel.com/2009/05/ciclo-de-vida-do-midlet.html>. Acesso em: 16 maio 2010.

JCRAFT. **JSch for J2ME**. Sendai, [2005]. Disponível em: <http://j2me.jsch.org/>. Acesso em: 18 maio 2010.

KUROSE, James F.; ROSS, Keith W. **Redes de computadores e a Internet**: uma abordagem top-down. 3. ed. São Paulo: Pearson Addison Wesley, 2006.

LAUDON, Kenneth C.; LAUDON, Jane P. **Sistemas de informação gerenciais**: administrando a empresa digital. 5. ed. São Paulo: Pearson Brasil, 2003. xx, 562 p.

LOPES, Raquel. V.; SAUVÉ, Jacques. P.; NICOLLETTI, Pedro. S. **Melhores práticas para gerência de redes de computadores**. Rio de Janeiro: Campus, 2003. 373 p.

MARTIN, Henrique. **Nokia compra a Symbian, que vira open source**. São Paulo, [2008]. Disponível em: <http://zumo.uol.com.br/2008/06/24/nokia-compra-symbian-vira-open-source/>. Acesso em: 07 maio 2010.

MATSUMOTO, Patrícia M. **SVG**: Scalable Vector Graphics. São Paulo, [2005]. Disponível em: <http://www.linux.ime.usp.br/~patty/mac499/tecnica/tecnologias/svg.html>. Acesso em: 11 abr. 2010.

MORAES, Marcelo. **A história do surgimento da linguagem Java**. Nova Iguaçu, [2009]. Disponível em: <http://www.marcelomoraes.com.br/conteudo/marcelo/java/historia_java.pdf>. Acesso em: 14 maio 2010.

MOREIRA, Daniela. **Smartphones levam produtividade e convergência ao mundo corporativo**. São Paulo, [2006]. Disponível em: <http://idgnow.uol.com.br/telecom/2006/08/08/idgnoticia.2006-08-07.1112897993/>. Acesso em: 02 maio 2010.

MORIMOTO, Carlos E. **Smartphones**: a história do Symbian. Porto Alegre, [2008]. Disponível em: <http://www.gdhpress.com.br/blog/historia-symbian/>. Acesso em: 26 mar. 2010.

MUCHOW, John W. **Core J2ME**: tecnologia e MIDP. São Paulo: Makron Books, 2004. 588 p.

NBSO. **Práticas de segurança para administradores de redes Internet**. São Paulo, 2003. Disponível em: <http://www.cert.br/docs/seg-adm-redes/seg-adm-redes.html>. Acesso em: 18 abr. 2010.

NEMETH, Evi; SNYDER, Garth; HEIN, Trent R. **Manual completo do Linux**: guia do administrador. 2. ed. São Paulo: Pearson Prentice Hall, 2007. 684 p.

NETO, Urubatan. **Dominando Linux Firewall Iptables**. Rio de Janeiro : Ciência Moderna, 2004. 98 p.

PÉRICAS, Francisco. A. **Redes de computadores**: conceitos e a arquitetura internet. Blumenau: EdiFURB, 2003. 158 p.

PRETTO, Carlos O. et al. Utilização de computação móvel para qualificação de rotinas de operação e manutenção de redes de distribuição. **Sba Controle & Automação**, Campinas, v. 17, n. 4, dez. 2006. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-17592006000400006&lng=en&nrm=iso>. Acesso em: 02 maio 2010.

RED HAT LINUX. **Red Hat Linux reference guide**. North Carolina, [2003]. Disponível em: <http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/ch-proc.html>. Acesso em: 12 mar. 2010.

REDE NACIONAL DE ENSINO E PESQUISA. **Introdução a gerenciamento de redes TCP/IP**. Rio de Janeiro, [1997]. Disponível em: <http://www.rnp.br/newsgen/9708/n3-2.html>. Acesso em: 02 maio 2010.

REZENDE, Denis A.; ABREU, Aline F. de. **Tecnologia da informação aplicada a sistemas de informação empresariais**: o papel estratégico da informação e dos sistemas de informação nas empresas. 2. ed. São Paulo: Atlas, 2001. 311 p.

SZTAJNBERG, Alexandre. **Gerenciamento de redes**: conceitos básicos sobre os protocolos SNMP e CMIP. Rio de Janeiro, [1996]. Disponível em: <http://www.gta.ufrj.br/~alexszt/ger/snmpcmip.html#sec1>. Acesso em: 15 abr. 2010.

VIVA SEM FIO. **Programas para Symbian**. Ponta Grossa, [2008]. Disponível em: <http://www.vivasemfio.com/blog/category/symbian_os/page/3/>. Acesso em: 12 maio 2010.