

**GERENCIAMIENTO DE SERVIDORES LINUX ATRAVÉS DE  
DISPOSITIVOS MÓVEIS COM SISTEMA OPERACIONAL  
SYMBIAN S60**

**Rafael Schleuss**

**Francisco Adell Péricas - Orientador**

# Roteiro da apresentação

- Objetivos do trabalho
- Fundamentação teórica
- Desenvolvimento
- Conclusões
- Extensões
- Apresentação do protótipo

# Objetivos do trabalho

- Motivação
- O objetivo do trabalho é desenvolver uma interface remota para gerenciamento e monitoração de servidores Linux utilizando dispositivos móveis dotados de sistema operacional Symbian

# Fundamentação teórica

- Gerenciamento de redes
  - ▣ atividade de administração
  - ▣ tarefa do administrador de redes
  - ▣ tolerância a falhas

# Fundamentação teórica

- Sistema operacional Linux
  - LSB
  - informações do kernel
  - firewall

# Fundamentação teórica

## □ Informações do kernel

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

```
[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
178697081
btime 1272751868
processes 385159
procs_running 1
procs_blocked 0
```

# Fundamentação teórica

---

- Dispositivos móveis
  - Symbian
  - J2ME

# Trabalhos correlatos

---

- LISP-CEL
- Aplicação web para gerenciamento de firewall Linux
- Webmin



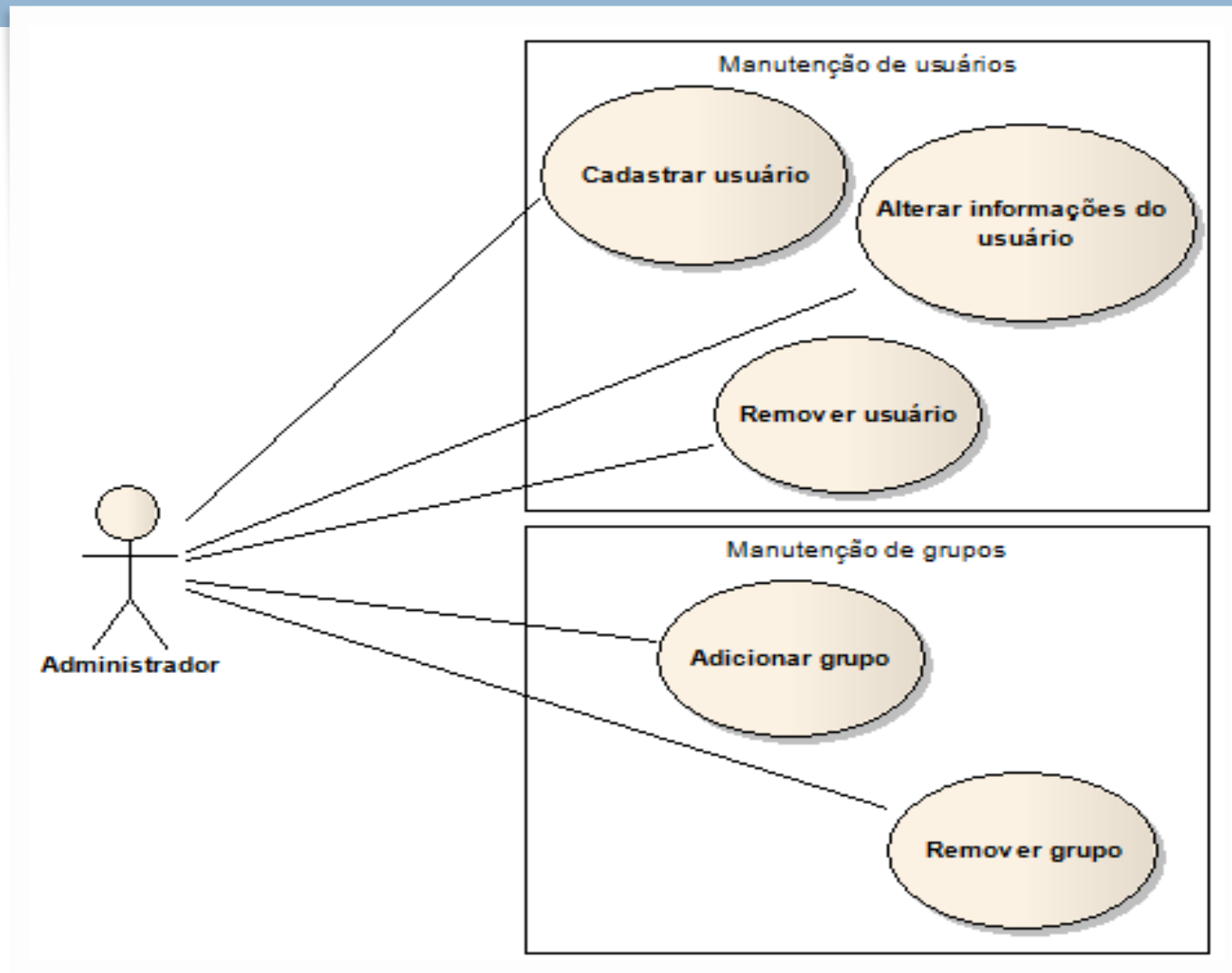
# Desenvolvimento - Requisitos

- Requisitos Funcionais
  - ▣ gerenciar usuários no servidor Linux
  - ▣ controlar o acesso aos recursos da rede
  - ▣ gerar estatísticas de recursos do servidor Linux
  - ▣ gerenciamento de serviços do servidor
  - ▣ alertas de segurança
  - ▣ utilizar o protocolo SSH para comunicação
  - ▣ chave simétrica para a autenticação do usuário

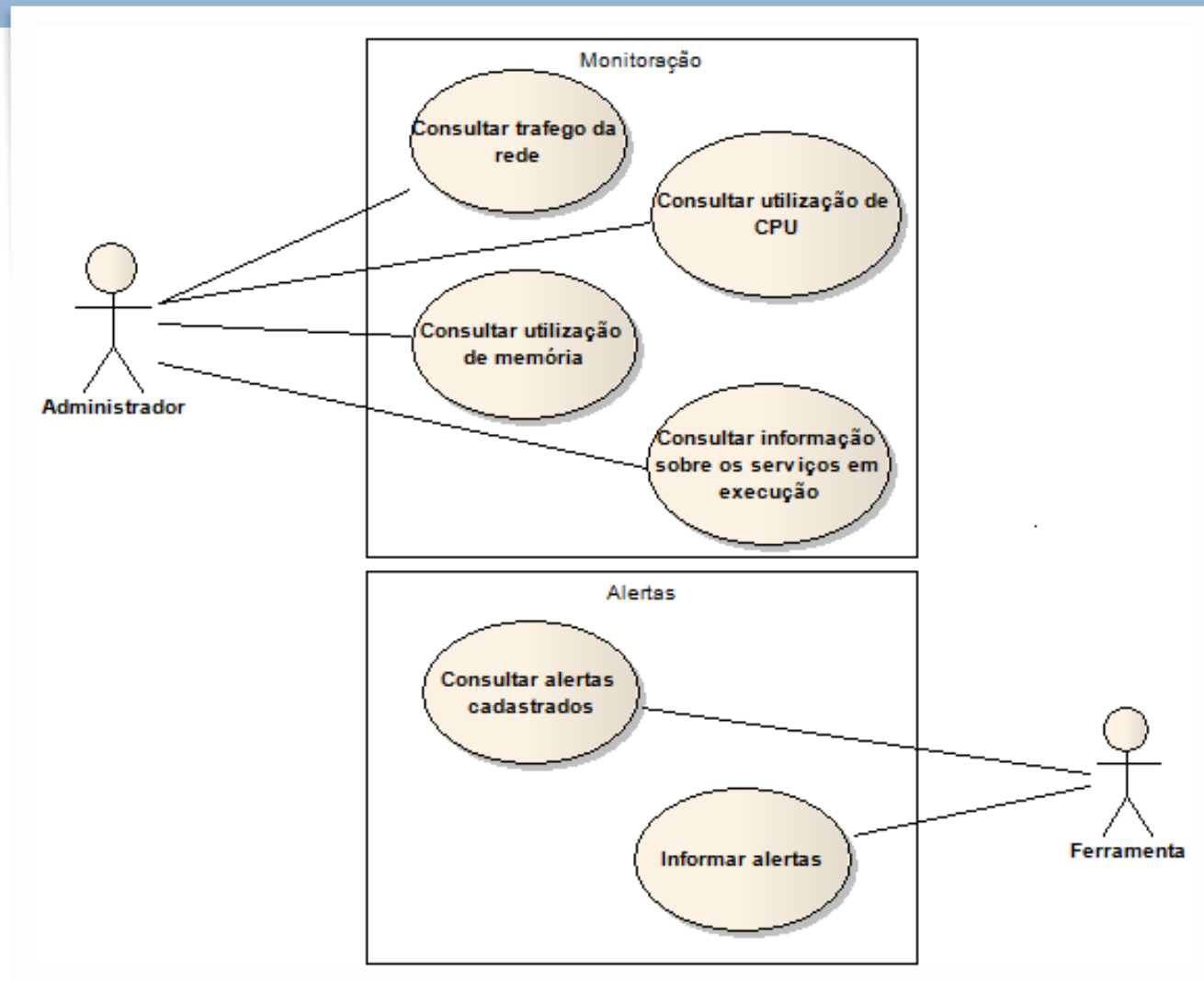
# Desenvolvimento - Requisitos

- Requisitos Não Funcionais
  - ▣ ser desenvolvido em linguagem Java
  - ▣ utilizar o ambiente de desenvolvimento Netbeans
  - ▣ utilizar análise orientada a objetos
  - ▣ interface intuitiva para usuários

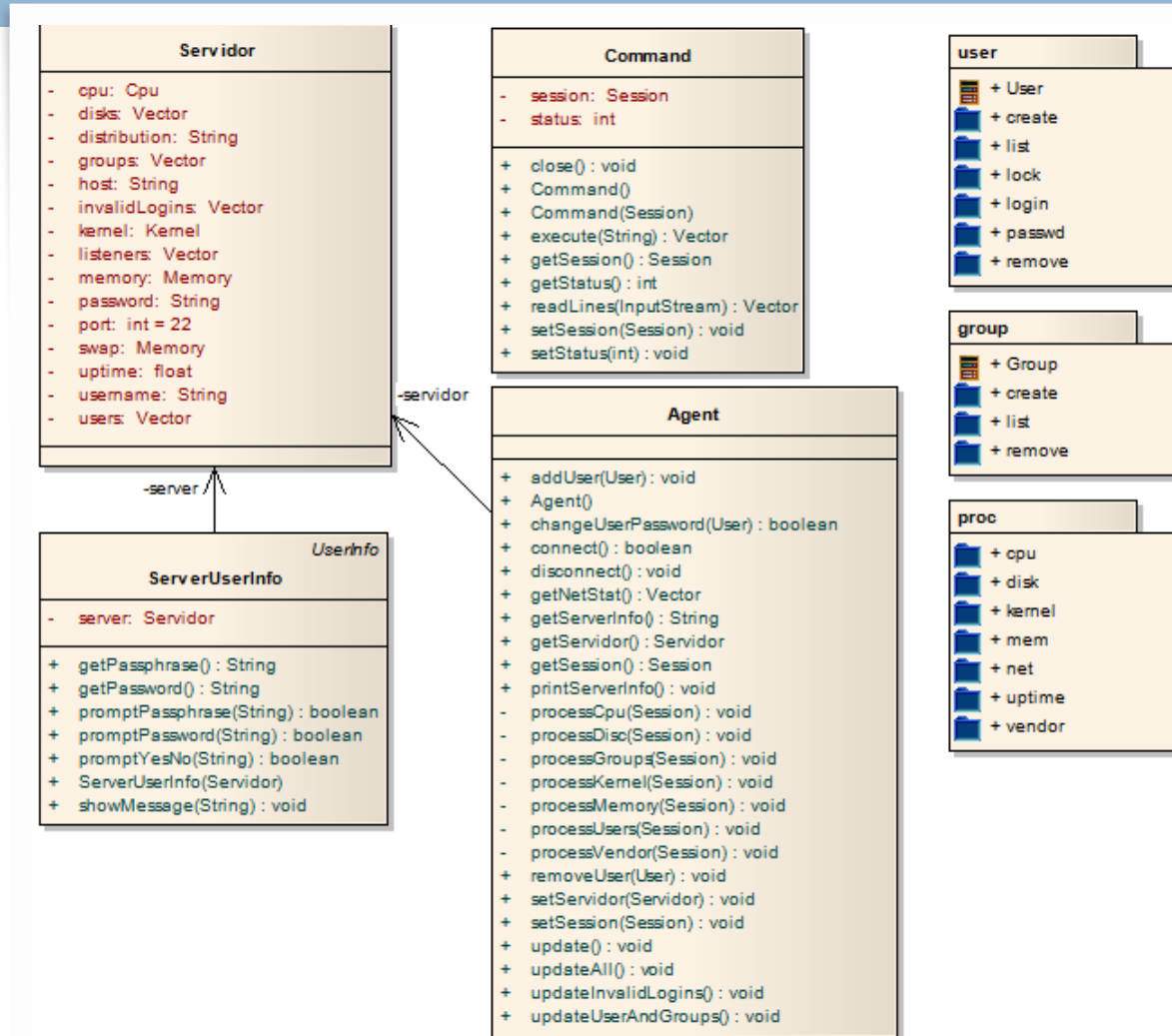
# Diagrama de casos de uso



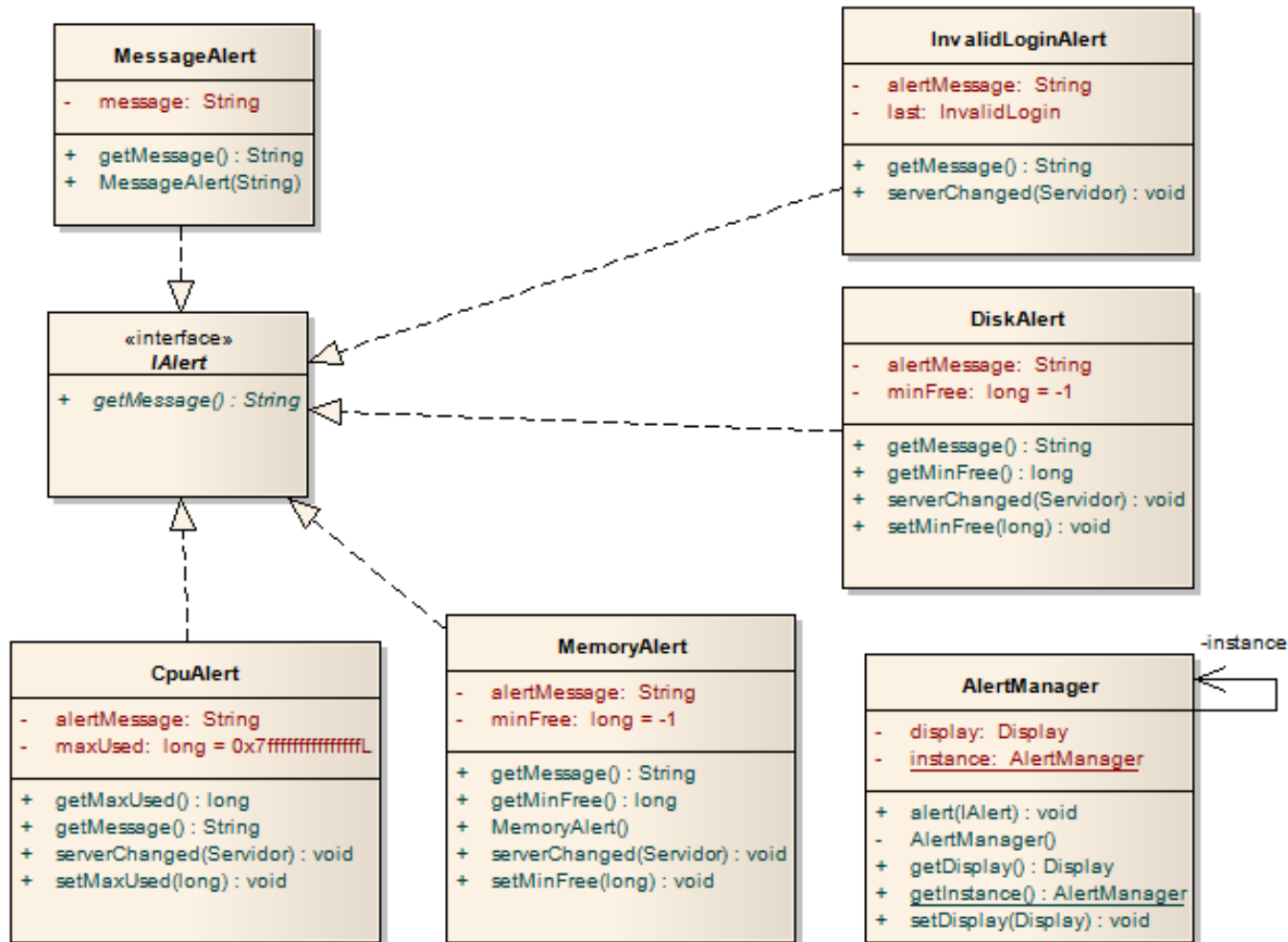
# Diagrama de casos de uso



# Diagrama de classes



# Diagrama de classes



# Código de conexão

```
// 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();
```

# Código de conexão

```
if(servidor.getKeyfile() != null) {
    Class c = this.getClass();
    StringBuffer b = new StringBuffer();
    InputStream is = c.getResourceAsStream(servidor.getKeyfile());
    int ch;
    while ((ch = is.read()) != -1) {
        b.append((char) ch);
    }
    IdentityMem id = new IdentityMem(b.toString(), "", jsch);
    id.setPassphrase(servidor.getPassword());
    jsch.addIdentity(id);
}
```



# Executar comando remoto

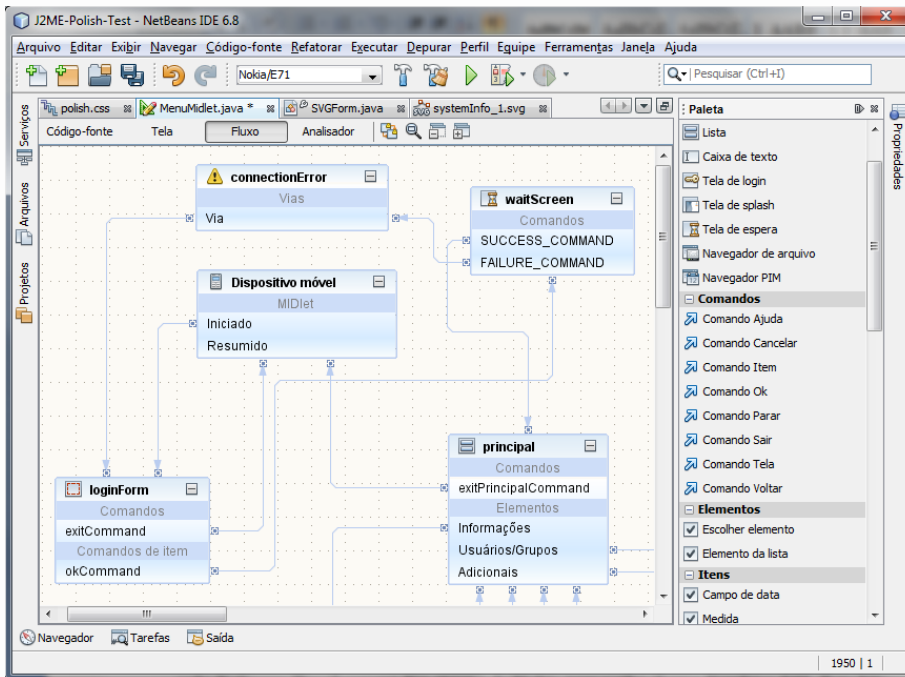
```
public Vector execute(String command) throws IOException, JSchException
{
    // abrir um canal de execucao de comandos
    Channel channel = getSession().openChannel("exec");
    channel.setInputStream(null);
    ((ChannelExec) channel).setErrStream(System.err);
    // informar comando
    ((ChannelExec) channel).setCommand(command);

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

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

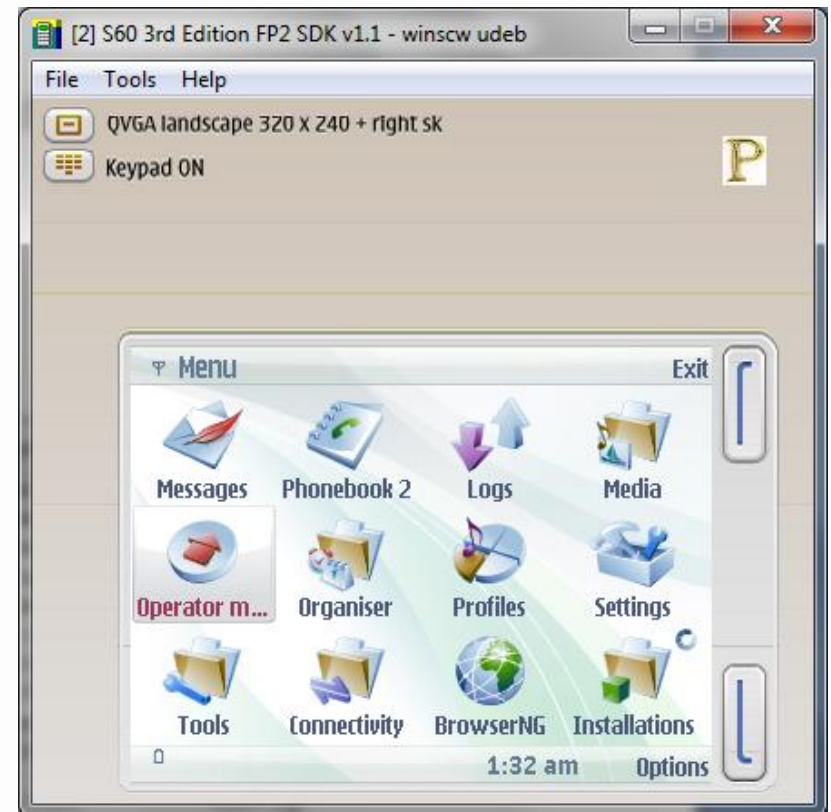
    // retornar linhas
    return result;
}
```

# Ferramentas



Netbeans IDE

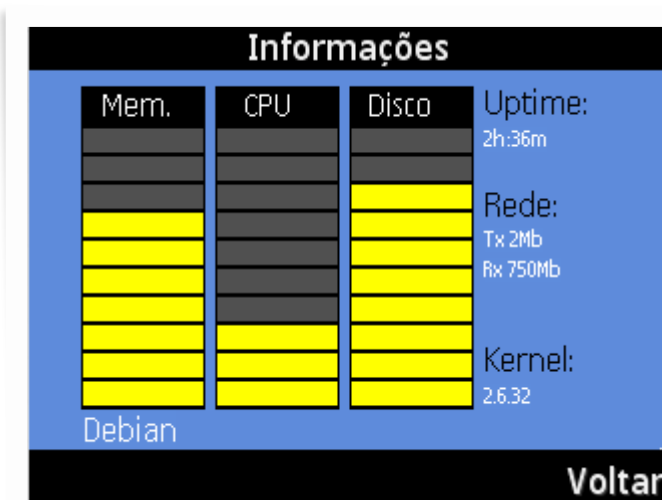
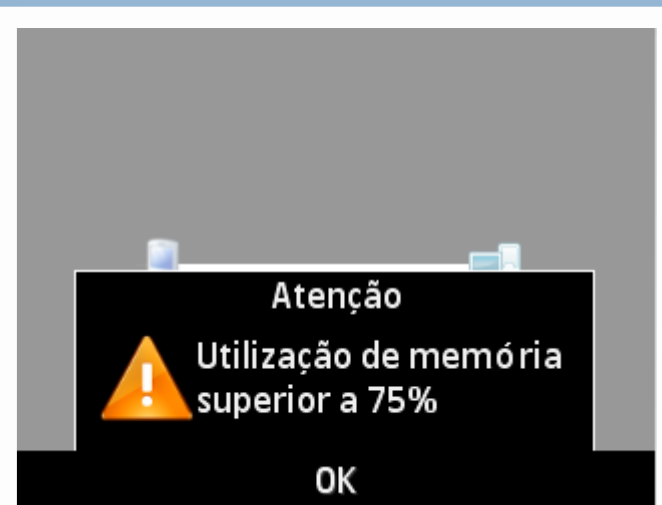
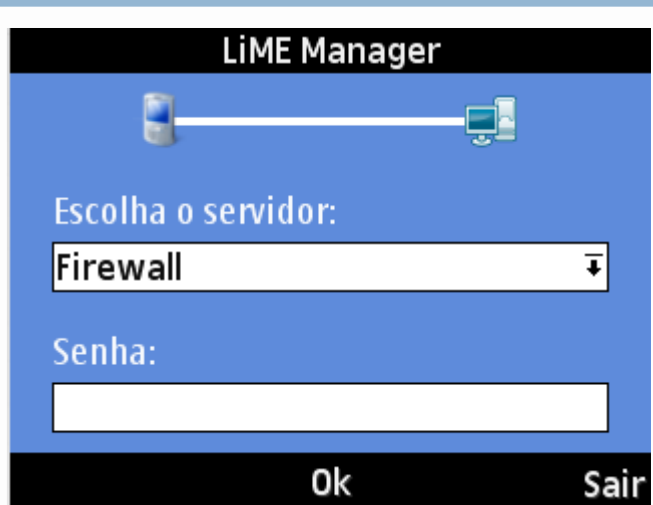
## Emulador Symbian S60



# Operacionalidade

## Requisitos

- j2me polish
- jsch



# Resultados

---

- Administração remota
- Mobilidade
- Interface simplificada

# Conclusão

---

- Benefícios e dificuldades
- Ferramentas de desenvolvimento
- Documentação
- Bibliotecas

# Extensões

- ❑ Interface touchscreen
- ❑ Descoberta de computadores por broadcast
- ❑ Cadastro de servidores e geração de chave pública/privada
- ❑ Enviar relatório de alarmes por e-mail
- ❑ Contatar usuários por e-mail

# Apresentação

