Mobile Command

Diego Armando Gusava

Orientador: Mauro Marcelo Mattos

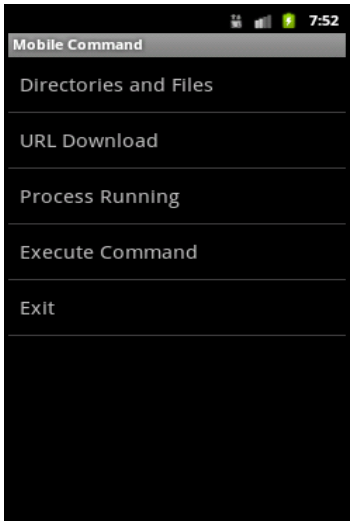
Roteiro

- Introdução
  - Objetivos
- Fundamentação teórica
- Desenvolvimento
- Implementação
- Conclusão
- Extensões

Introdução

O que me motivou?

Solução proposta



Objetivos

- Disponibilizar um protocolo de comunicação entre um desktop e um smartphone.
- Disponibilizar uma aplicação no *smartphone* para acesso a comandos do sistema operacional a ser controlado

Fundamentação teórica

Fundamentação Teórica

- Protocolo TCP/IP e TLS v1
- Modelo Cliente/Servidor
- Shell Linux
  - Executar comandos
  - Visualizar e finalizar processos
- Plataforma Android

Plataforma Android

- Principais componentes
  - Activity
    - Tela da aplicação
    - Controlar eventos
    - XML responsável por desenhar a interface gráfica
  - Intent
    - Mensagem
  - Service
    - Serviço em segundo plano

Trabalhos correlatos

- Resmo
- PhoneMyPc
- Ignition

Desenvolvimento

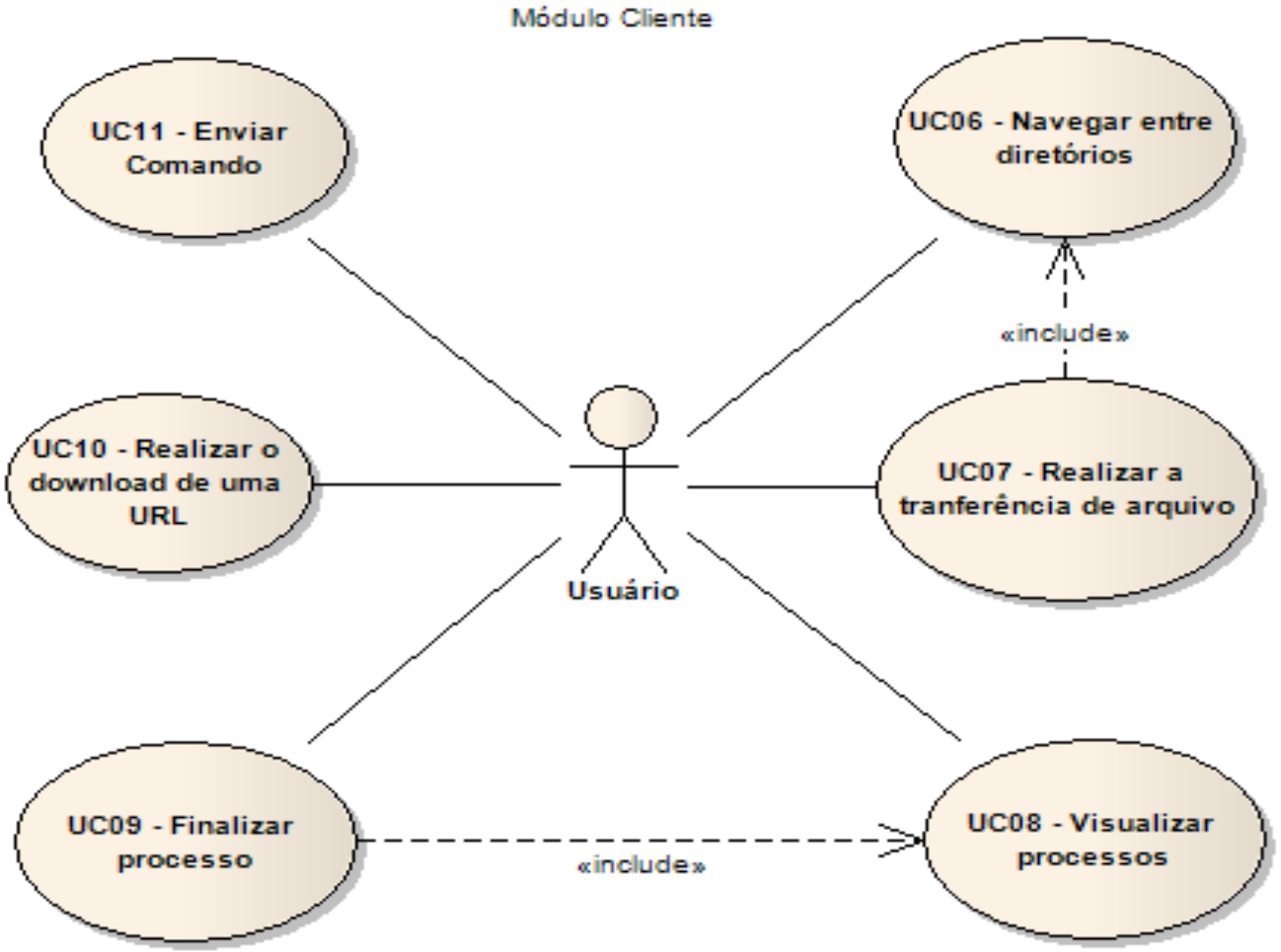
Requisitos

- RF01 - Permitir que o software cliente navegue entre diretórios e arquivos do sistema operacional no qual o software servidor está instalado.
- RF02 - Permitir que o software cliente envie comandos para o software servidor.
- RF03 - Permitir que o software servidor realize *downloads* de dados da internet através de comandos enviados pelo software cliente.
- RF04 - Permitir que o software servidor execute e finalize aplicativos no *desktop* através de comandos enviados pelo software cliente.
- RF05 - Permitir que o software cliente visualize os aplicativos em execução no sistema operacional onde o software servidor está instalado.

Requisitos

- RNF01 - Oferecer ao software cliente uma lista amigável de serviços que podem ser executados no software servidor
- RNF02 - Deve ser implementados na linguagem Java.
- RNF03 - Deve ser implementado usando o ambiente de desenvolvimento Eclipse.
- RNF04 - Dados sigilosos do usuário devem ser criptografados.
- RNF05 - O sistema operacional no qual o software cliente está instalado deve rodar no sistema operacional Android.
- RNF05 - O sistema operacional no qual o software servidor está instalado deve rodar no sistema operacional Linux .

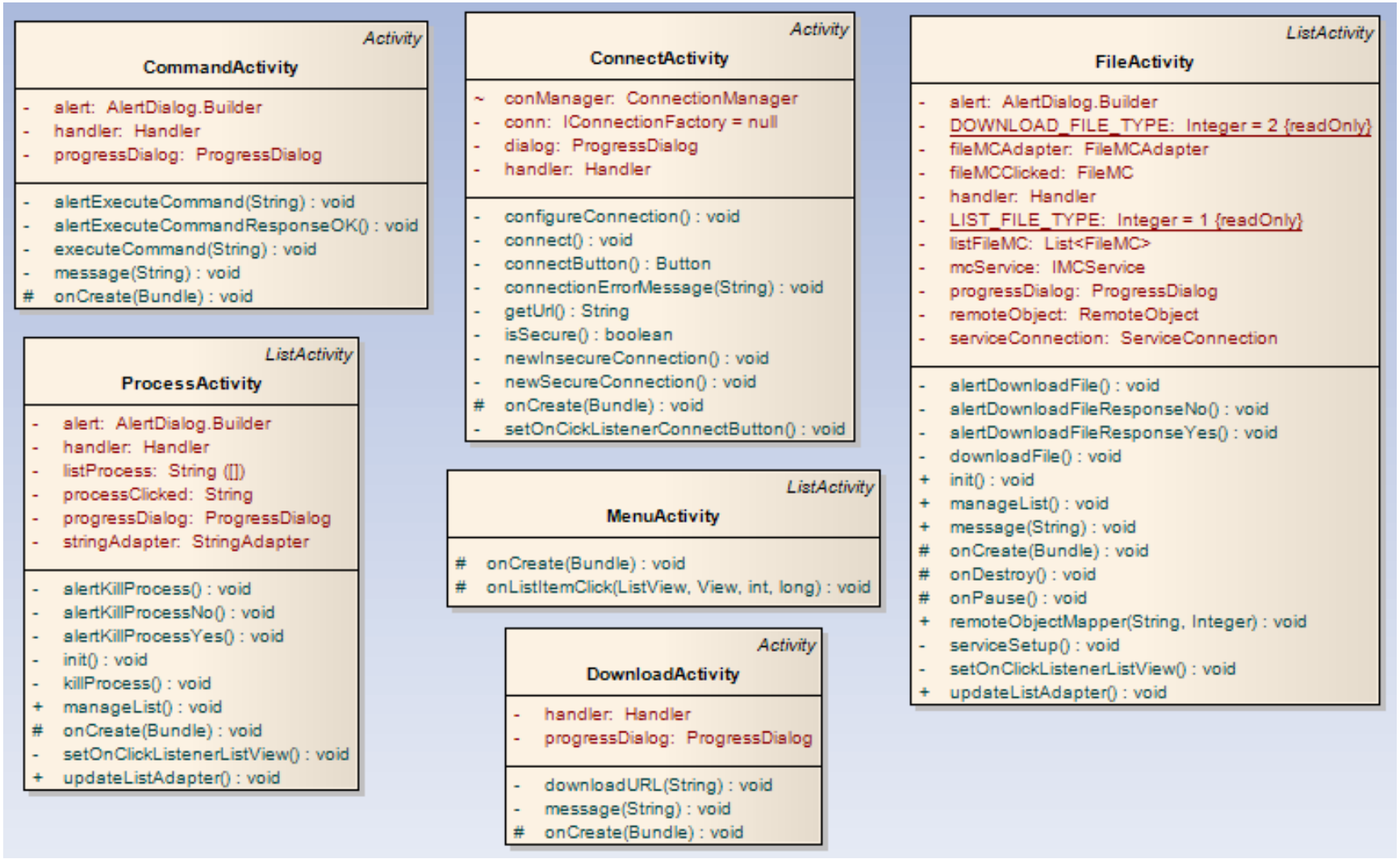
Módulo Cliente



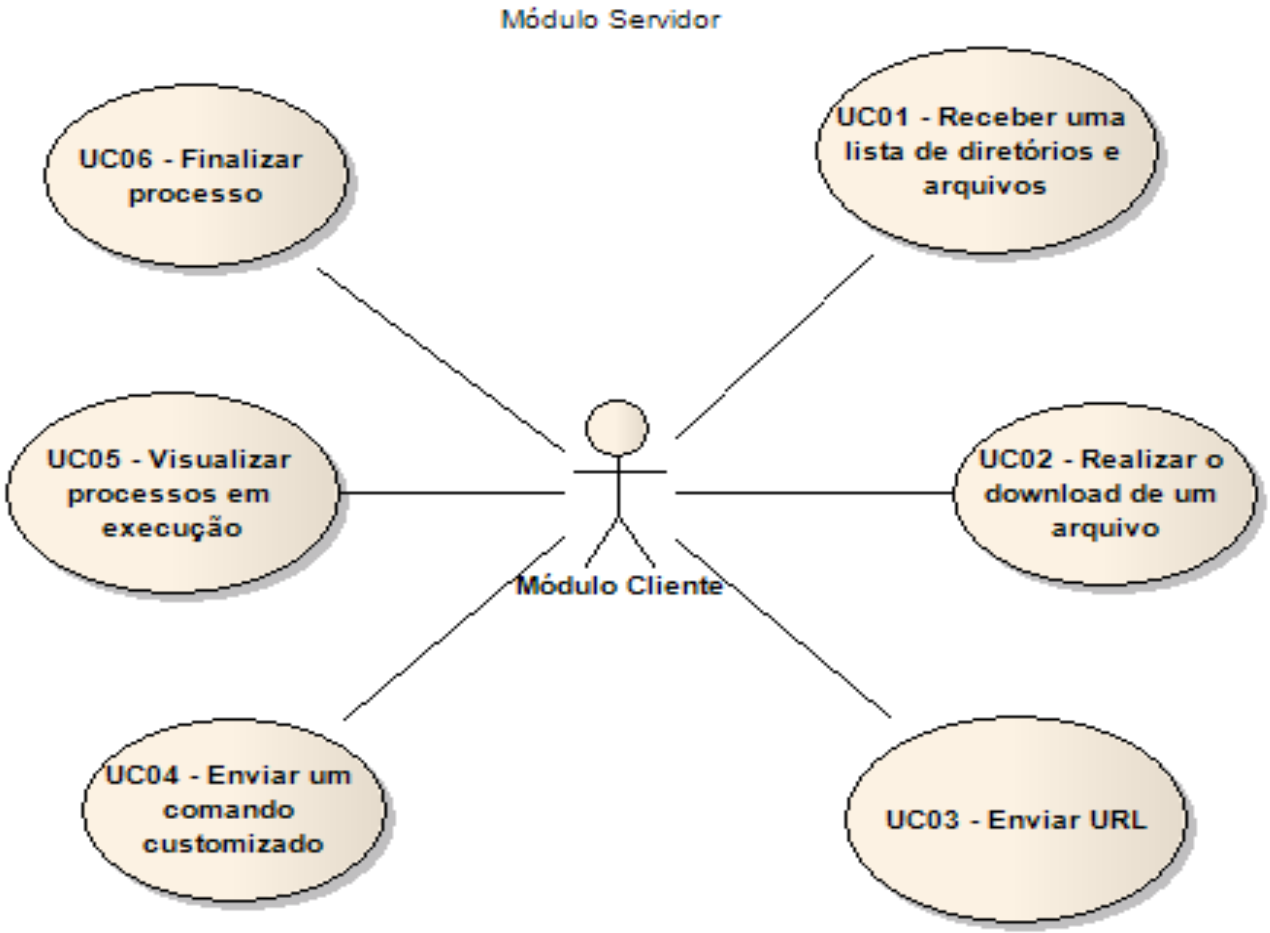
Módulo Cliente

- 8 pacotes
- Principais pacotes:
  - Activity
    - Responsáveis por interagir com o usuário
  - Connection
    - Estabelecer conexão com o servidor
  - Manager
    - Envia e trata as requisições ao módulo servidor
  - Service
    - Criar um serviço em segundo plano
  - Model
    - Objeto remoto : FileMC e o RemoteObject

Módulo Cliente



Módulo Servidor



Módulo Servidor

- 4 pacotes
- Pacotes Principais
  - Connection
    - Estabelecer a conexão
  - Model
    - Objetos remotos
  - Helper
    - Gerenciar as requisições

Módulo Servidor

- Escutar requisições de abertura de conexão
  - SecureConnectionFactory
  - InsecureConnectionFactory

```
@Override
public void run() {
    while(true){
        Connected connected = new Connected(newConnection());
        new Thread(connected).start();
    }
}
```

Módulo Servidor

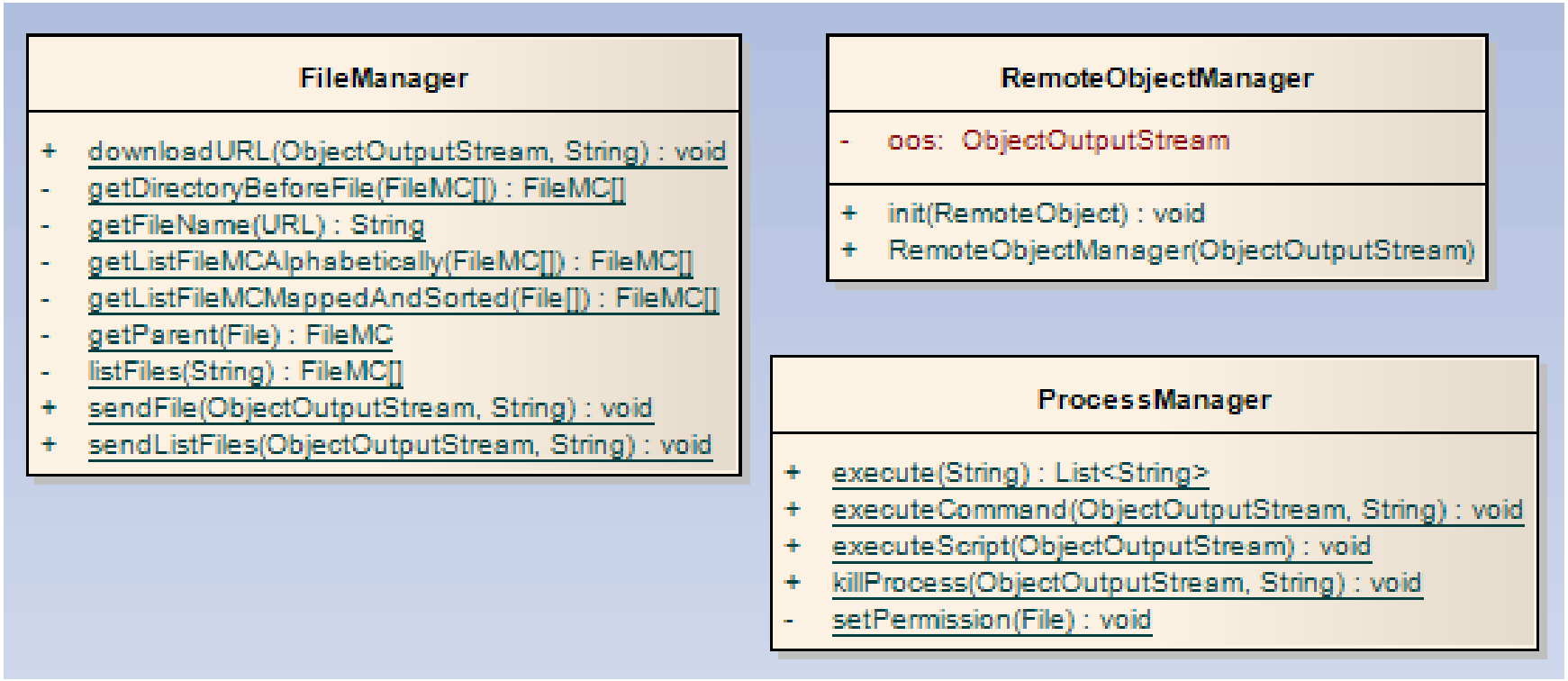
- Connected
  - Aguardando novas requisições

```java
@Override
public void run() {
    try {
        RemoteObject remoteObject = (RemoteObject)ois.readObject();
        RemoteObjectManager roManager = new RemoteObjectManager(this.oos);
        while(true){
            roManager.init(remoteObject);
            remoteObject = (RemoteObject)ois.readObject();
        }
    }catch (SocketTimeoutException ste){
        System.out.println("TIME OUT");
        this.close();
    }catch(java.io.EOFException eof){
        System.out.println("Client close connection");
    }catch (IOException e) {
        e.printStackTrace();
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

Módulo Servidor

```
public void init(RemoteObject ro){
        Integer value = ro.getType();
        final int DIRECTORY_LIST = 1;
        final int SEND_FILE = 2;
        final int DOWNLOAD_FILE = 3;
        final int PROCESS_RUNNING = 4;
        final int KILL_PROCESS = 5;
        final int EXECUTE_COMMAND = 6;
        switch (value) {
            case DIRECTORY_LIST: {
                FileManager.sendListFiles(oos, ro.getCommand());
                break;
            }
            case SEND_FILE: {
                FileManager.sendFile(oos, ro.getCommand());
                break;
            }
            case DOWNLOAD_FILE: {
                FileManager.downloadURL(oos,ro.getCommand());
                break;
            }
            case PROCESS_RUNNING:{
                ProcessManager.executeScript(oos);
                break;
            }
            case KILL_PROCESS:{
                ProcessManager.killProcess(oos,ro.getCommand());
                break;
            }
            case EXECUTE_COMMAND:{
                ProcessManager.executeCommand(oos,ro.getCommand());
                break;
            }
            default:{
                break;
            }
        }
}
```

Módulo Servidor



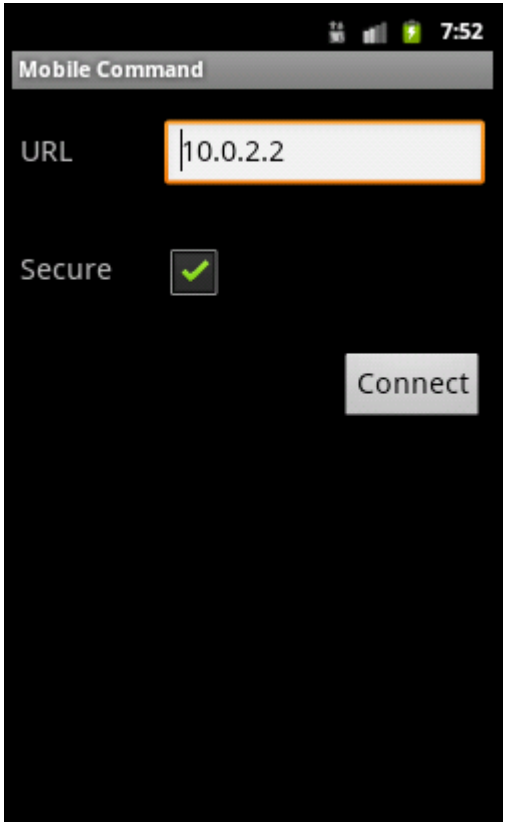
Implementação

Ferramentas Utilizadas

- Linguagem: Java
- IDE: Eclipse
- Android SDK
- *Android Development Tools (ADT)*

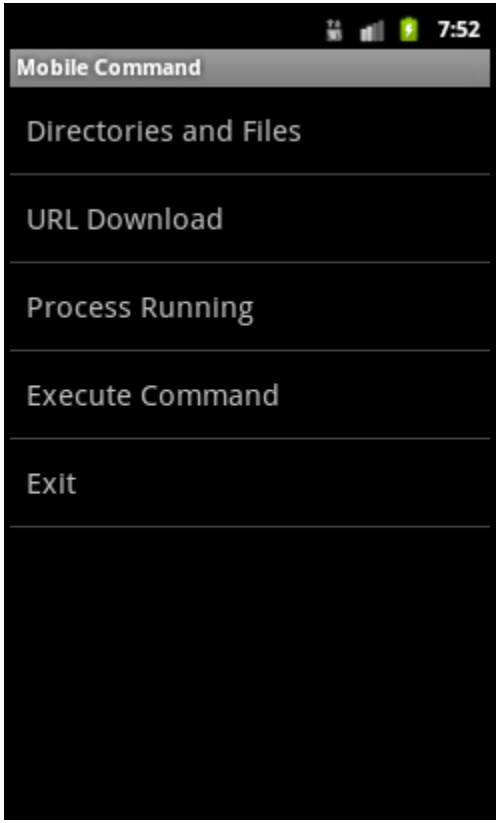
Operacionalidade

Tela: formulário de conexão



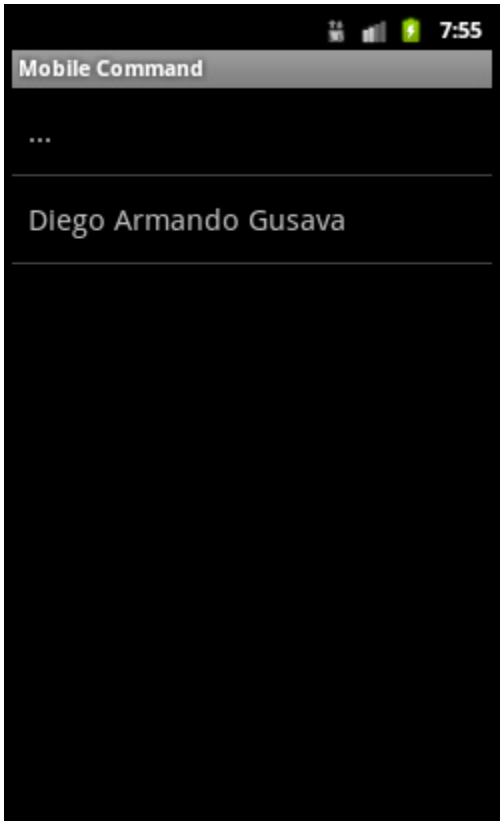
Operacionalidade

Tela: menu



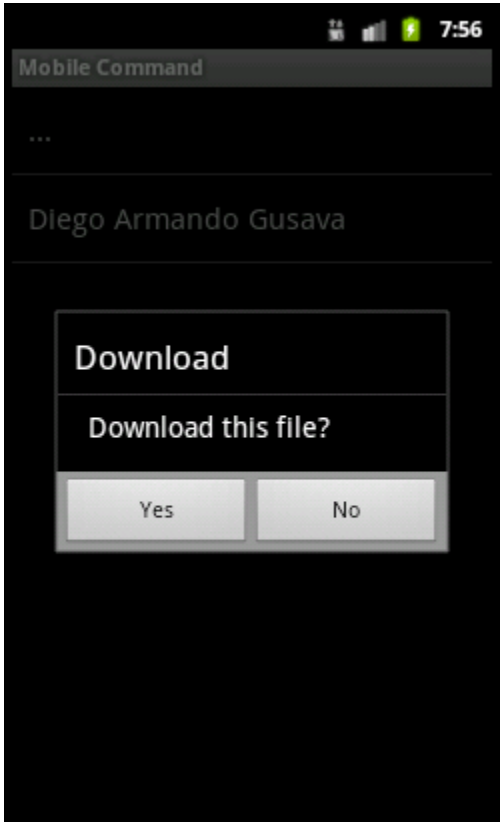
Operacionalidade

Tela: Diretórios e arquivos



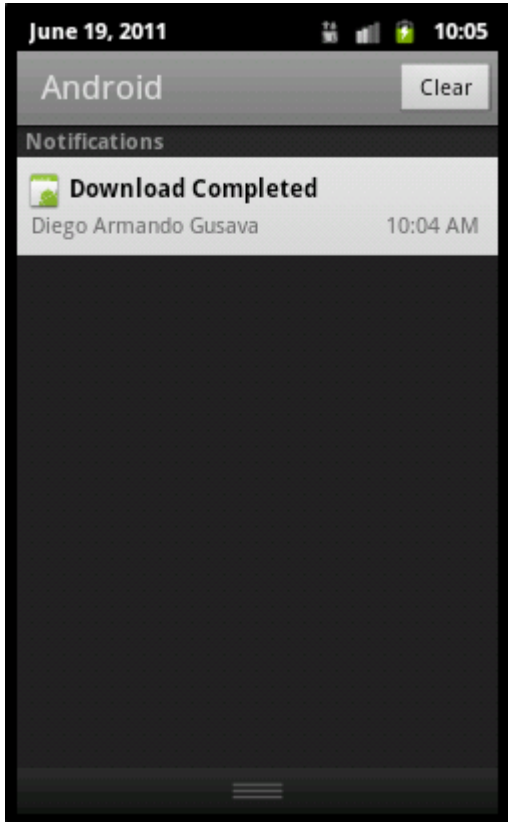
Operacionalidade

Tela: Download arquivo



Operacionalidade

Tela: Notificação de download finalizado



Resultados e discussão

Funções	**Linux**	**Windows**	**Mac**
Directories and Files	X	X	X
URL Download	X	X	X
Process Runing	X		
Execute Command	X		

Resultados e discussão

Funções	**MobileCommand**	**Ignition**	**ResMo**	**PhoneMyPC**
Acesso a área de trabalho do computador		X		X
Controle do mouse do computador		X		X
Executar comandos remotos	X	X	X	X
Visualizar diretórios e arquivos	X	X	X	X
Download de uma URL	X	X	X	X
Finalizar processos	X	X	X	X
Transferência de arquivo	X			
Baixa transferência de dados	X		X	

Conclusão

Conclusão

- Foi possível estudar a plataforma Android
- Os objetivos foram atingidos
  - Visualizar arquivos
  - Download de arquivo
  - Visualizar processos
  - Executar e fechar aplicativos
  - Servidor fazer o download de uma URL

Extensões

Extensões

- Controlar o módulo cliente através de comandos enviados pelo módulo servidor.
- Disponibilizar funções para o *smartphone* servir como um controle remoto do servidor, podendo abrir, executar e parar vídeos, músicas, etc.
- Disponibilizar o módulo cliente para os *smartphones* que possuam outro sistema operacional instalado, além do Android.