



UNIVERSIDADE LUSÓFONA
de Humanidades e Tecnologias
Humani nihil alienum

ActivoBank
by Millennium

Relatório do Trabalho de Final de Curso

Do Curso de

Licenciatura em Engenharia Informática (LEI)

- Pagamentos Multibanco Confirmados em *Smartphones* -

Realizado por:

Miguel Catalino
Nº20094915

Índice

Introdução	3
Objectivo do Trabalho	3
Descrição do Problema	3
Implementação do Problema	4
Use Cases	5
Use Case nº1 – Activar/Desactivar Serviço	5
Use Case nº2 – Efectuar Pagamento Multibanco	5
Use Case nº3 – Efectuar Primeira Autenticação na Aplicação	5
Diagrama de Classes	6
Implementação	7
Testes Efectuados	10
Conclusão	11
Bibliografia e Referencias	11
Anexos	12

1. Introdução

O trabalho de final de curso representa o culminar de três anos de aprendizagem que a Engenharia Informática proporcionou ao longo de todos os anos lectivos na faculdade.

Como tal, a proposta feita pelo *ActivoBank* – Pagamentos por Multibanco confirmados via *Smartphone* – acabou por ser a escolhida para estudo e implementação.

Tal como é sugerido no título da proposta, deve-se implementar um sistema que consiga fazer a gestão de pedidos feitos via multibanco, de forma a que o sistema sege capaz de confirmar ou recusar todos os pedidos feitos por multibanco. Pretende-se então que tanto o sistema *SIBS* e multibanco em geral fiquem implementados com o actual sistema, sendo apenas pedido uma solução que apenas altere o sistema do *ActivoBank*.

Este projecto foi realizado sob a orientação do Prof. Sérgio Guerreiro.

1.1. Objectivo do Trabalho

Este trabalho tem como objectivo estudar as possíveis possibilidades de pagamento via *Smartphone*, verificar o público-alvo para cada implementação e, desenhar e desenvolver todo o sistema tanto do *ActivoBank* como do *Smartphone*.

Sendo o *Smartphone* um equipamento pessoal e intransmissível, pode-se pensar que vai ser um sistema tão ou mais seguro que o actual sistema multibanco.

1.2. Descrição do Problema

No início deste projecto teve-se de estudar primeiro quais as melhores implementações para este tipo de problema, neste caso ou um sistema implementado directamente no *Smartphone* que comunica activamente com o sistema do *ActivoBank*, ou um sistema de criação de novos *PIN's* e envio dos mesmos por *SMS*.

As vantagens de ter um sistema por *SMS* são apenas o facto de o *PIN* estar em constante mudança, nunca mantendo o mesmo código, mas por outro lado, as desvantagens acabam por se sobrepor, no caso de se perder o *Smartphone* ao mesmo tempo que o cartão multibanco, qualquer pessoa mal-intencionada poderia facilmente descobrir o *PIN* do mesmo e até mesmo o atraso da recepção da *SMS* poder meter todo o processo em questão.

Por outro lado, o sistema directamente implementado no *Smartphone*, acaba por dar mais segurança e portabilidade a todo o sistema. Este sistema adiciona uma nova camada de segurança ao sistema actual pois o utilizador tem de desbloquear a aplicação com recurso a um *PIN* da própria aplicação. Desta maneira o próprio pagamento pode ser não presencial, sendo que basta estar um portador do cartão no acto da compra, sendo que o cliente pode aceitar/recusar pagamentos remotamente, mesmo que o *PIN* inserido no terminal não sege o correcto. A grande desvantagem poderá ser atraso na entrega de dados o que poderá originar um *timeout* por parte da rede *SIBS*.

1.3. Implementação do Problema

Em reunião com o orientador do projecto decidiu-se que iria ser implementado a solução da aplicação no *Smartphone*, sendo assim, a solução passou pela implementação em *Android* da aplicação e em *JAVA* do sistema *ActivoBank*.

Ao estudar a implementação deste sistema, verificou-se um conjunto de particularidades:

- O utilizador pode não estar presente durante o acto de pagamento - evitando assim conceder o *PIN* do cartão a outros indivíduos;
- O utilizador poderá activar/desactivar o serviço quando assim desejar;
- Mesmo com o serviço activo o utilizador pode confirmar o pagamento no terminal, inserindo o *PIN* correcto do cartão, podendo evitar atrasos ou outros contratempos que possam existir na rede do *Smartphone*;
- Ao recusar no *Smartphone* o pagamento, o pagamento deve de ser recusado mesmo que o utilizador insira correctamente o *PIN* do cartão no terminal.

Ao iniciar a implementação do sistema foi necessário também criar dois sistemas de simulação, um para o sistema actual *SIBS* e outro para o terminal multibanco, ambos recorrendo igualmente a linguagem *JAVA*.

Todas as aplicações desenvolvidas foram projectadas para serem usadas em locais diferentes, apesar dependerem umas das outras, falam todas entre si via rede/internet.

O *timeout* criado pelo sistema *SIBS* está implementado e presente em todas as aplicações podendo todo o processo ser interrompido pelo mesmo.

Todos os sistemas usam *MySQL* como motor de base de dados e os sistemas da *SIBS* e do *ActivoBank* necessitam do *driver MySQL* correspondente para poderem funcionar correctamente.

2. Use Cases

2.1. Use Case nº1 – Activar/Desactivar o Serviço

O utilizador abre a aplicação, desbloqueia a aplicação com recurso ao seu *PIN*. Clica no botão designado para activar/desactivar o serviço, o *Smartphone* envia um código ao sistema do *ActivoBank* que deverá proceder á activação/desactivação do serviço.

2.2. Use Case nº2 – Efectuar Pagamento Multibanco

O utilizador insere o cartão no terminal multibanco, o terminal envia uma mensagem de pagamento á *SIBS* que envia um pedido ao *ActivoBank*, este verifica se o cliente tem o serviço activo. Se o cliente tem o serviço activo, o serviço envia um pedido de *PIN* á *SIBS* e simultaneamente envia para os dados de pagamento para o *Smartphone*.

Caso o cliente não tenha o serviço activo, basta inserir o código *PIN* e o sistema de pagamento não se altera. Caso tenha o serviço activado, o *Smartphone* do cliente vai alertá-lo de um novo pagamento, este deve de desbloquear a aplicação com o seu *PIN* e o *Smartphone* vai mostrar os dados do pagamento, logo de seguida o utilizador deve de escolher a opção de pagamento ou recusa.

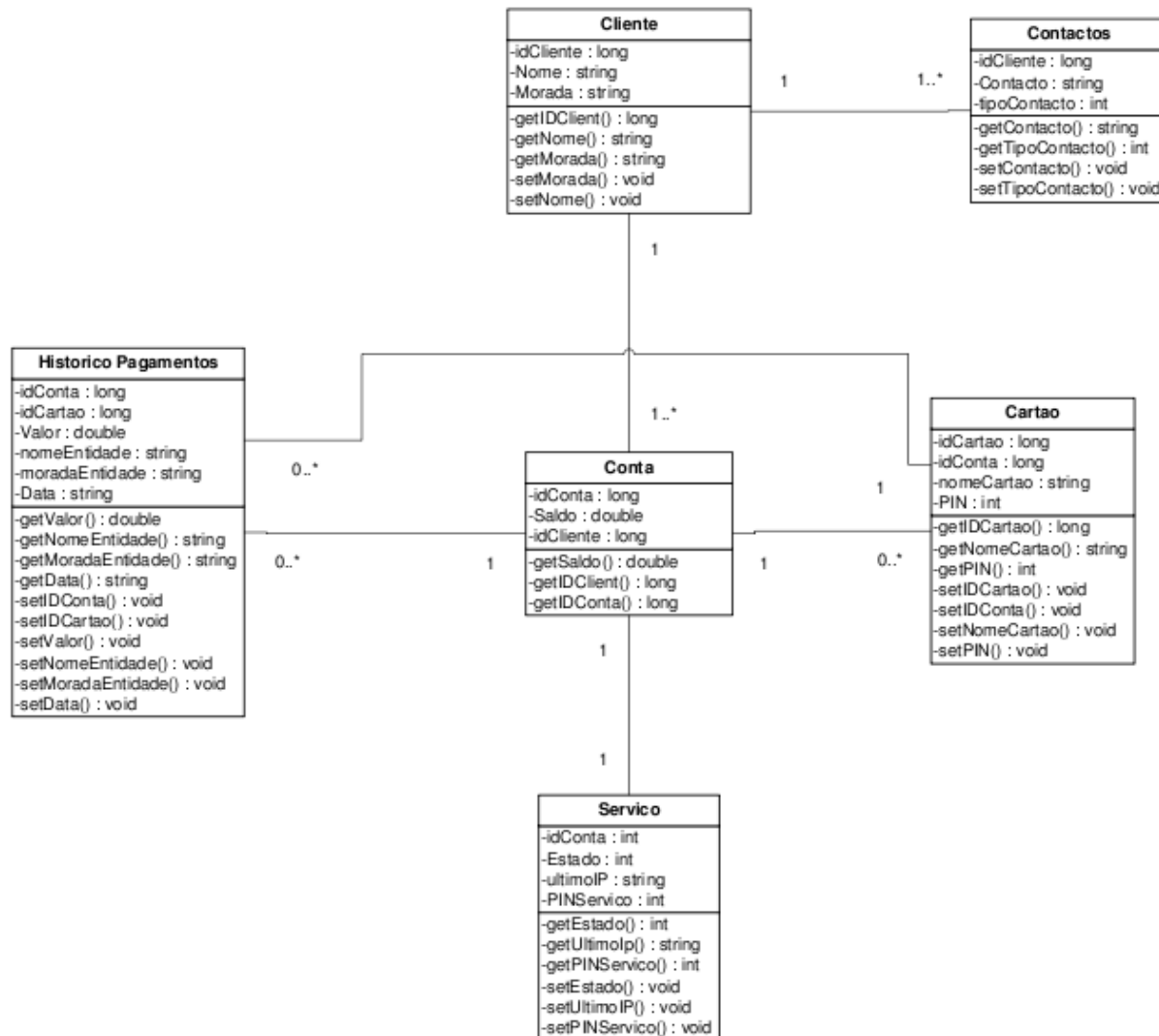
O sistema vai aguardar tanto pela confirmação/recusa do pagamento do *Smartphone* e pelo *PIN* pedido. Se não obtiver resposta do *Smartphone*, este verifica se o *PIN* está correcto e continua o pagamento normal, se tiver obtido resposta do *Smartphone*, o sistema pode descartar o *PIN* recebido e continuar com o sistema de pagamento normal mas com a resposta do *Smartphone*.

2.3. Use Case nº3 – Efectuar Primeira Autenticação na Aplicação

O utilizador abre a aplicação, esta vai pedir os dados de conta, número de conta e o *PIN* do serviço. Ao inserir o *Smartphone* vai comunicar com o sistema do *ActivoBank* e este vai verificar a autenticidade dos dados inseridos e vai retornar a resposta ao *Smartphone*.

Se o *ActivoBank* aceitar a autenticação, o *Smartphone* vai guardar as credenciais que são enviadas pelo sistema para que em futuros acessos o *Smartphone* possa pedir apenas o *PIN* de acesso.

3. Diagrama de Classes



No diagrama de classes estão representadas as classes Conta, Serviço, Cartão, Cliente, Histórico Pagamentos e Contactos.

Pode-se dizer que a classe Conta é uma Super-Classe e que as classes Cliente, Cartão Serviço e Histórico Pagamentos são Sub-Classes, estando assim presente o conceito de herança. Esta situação acontece pois todas as essas classes estão associadas a uma conta.

No caso da classe Contactos está apenas dependente da classe Cliente por ser apenas necessário o `idCliente` para saber quais os contactos do mesmo.

A classe Histórico Pagamentos está igualmente dependente da classe Cartão, por desta depender o número do cartão que vai estar associado ao pagamento em si.

4. Implementação

O passo inicial foi fazer a modelação do sistema do *ActivoBank* e do sistema para *Android*. Pensou-se no tipo de comunicação que ia ser efectuada e na linguagem para implementar o sistema. Decidiu-se então que a linguagem ia ser *JAVA* para todas as aplicações e que se ia usar o *MySQL* como motor de base de dados.

Logo desde o início da implementação desta resolução ao problema, foi necessário desenvolver dois simuladores, um para o terminal multibanco (POS) e outro para o sistema *SIBS*, a sua implementação foi baseada nos requisitos dados pelo *ActivoBank*.

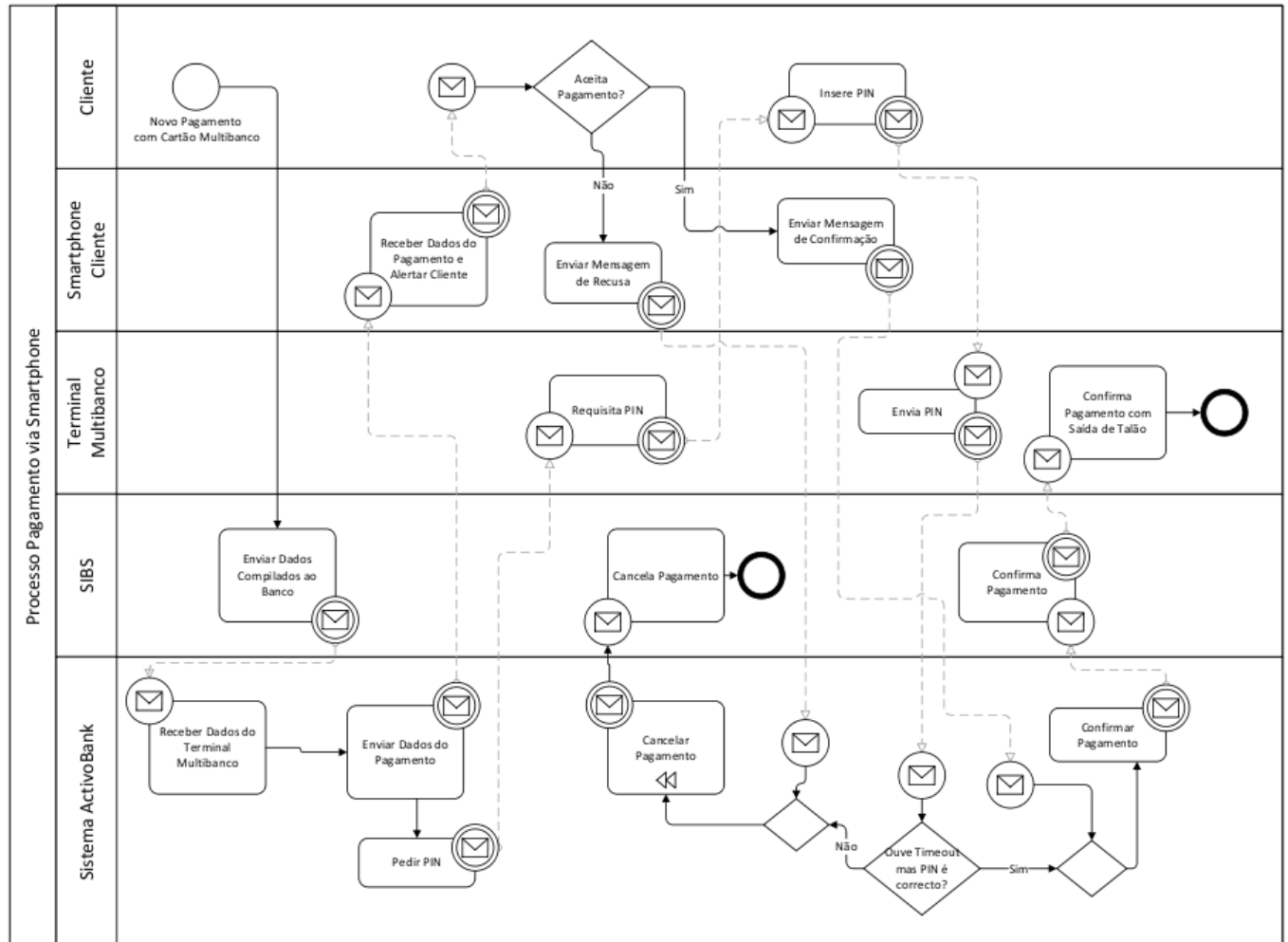
As aplicações comunicam entre si através da tecnologia *Socket*, já oferecida pela linguagem. O *timeout* implementado actualmente pelo sistema *SIBS* está presente em todas as aplicações, podendo até cancelar toda a operação de pagamento através do mesmo.

É na aplicação do *ActivoBank* que vão estar todas as definições, histórico e opções guardadas, o *Smartphone* vai apenas importar e guardar no local para rápida consulta.

Caso o serviço não se encontre activo, essa informação vai estar no sistema do banco, passando assim todas as definições e decisões para o *ActivoBank*.

Um dos grandes objectivos e desafios desta solução passava por ter um sistema que pudesse operar sob qualquer tipo de pagamento – no Terminal Multibanco ou no *Smartphone* – logo foi necessário implementar um sistema que autonomamente pudesse mudar o estado do pagamento – de confirmação no *Smartphone* para confirmação no Terminal Multibanco – podendo assim evitar conflitos ou atrasos na rede, evitando assim que o utente final possa por o sistema em questão.

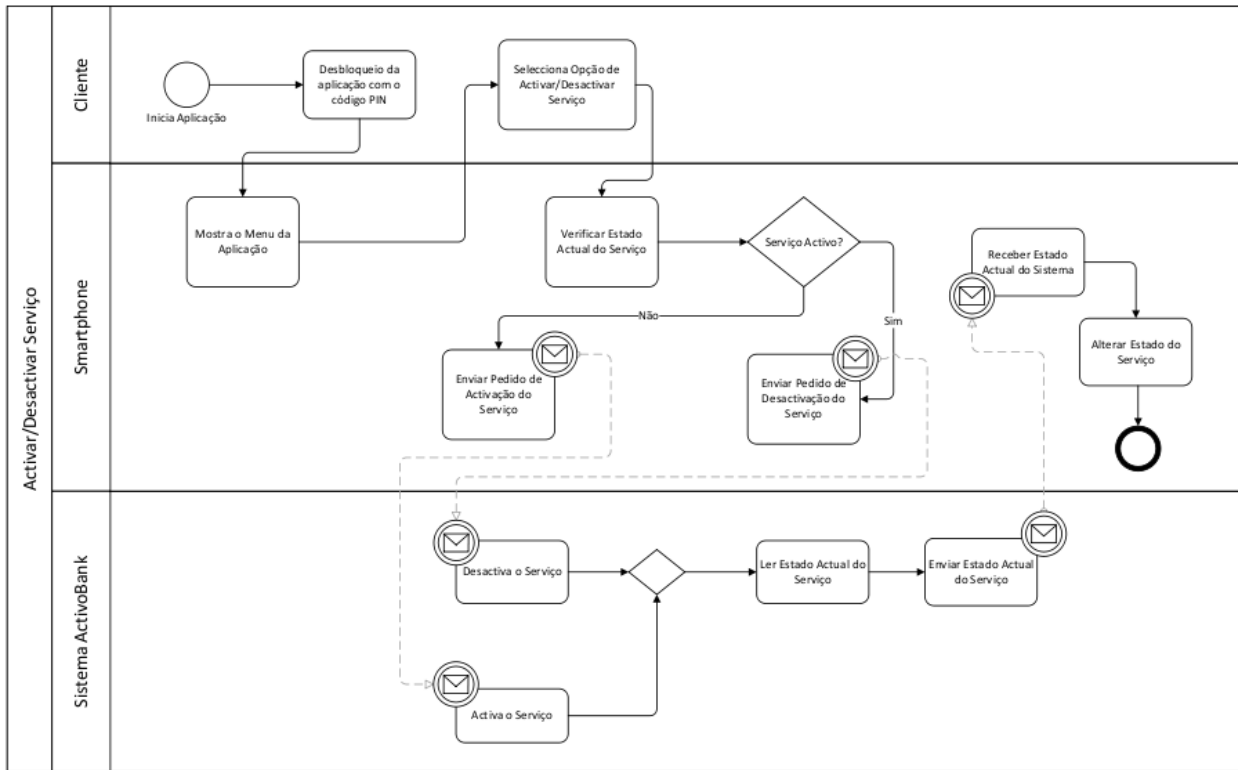
Desta forma o sistema tem de poder operar em conjunto com o Terminal Multibanco, ou seja, se o utilizador tiver o serviço de pagamento pelo *Smartphone* activo, este deve de poder efectuar o pagamento no Terminal Multibanco apenas, inserindo assim o *PIN* correcto do cartão, se o sistema não receber a resposta dada pelo *Smartphone*, este deve de continuar com o pagamento com o *PIN* inserido pelo cliente, como se pode ver no diagrama abaixo (versão reduzida).



Como se pode ver no diagrama acima, o cliente tem o sistema de pagamento por *Smartphone* activo, mas ao fazer um novo pagamento o sistema vai verificar se existe algum tipo de *timeout* por parte do *Smartphone*, se não houver, o sistema vai usar a resposta recebida para recusar ou aceitar a transacção. No caso de o sistema detectar um *timeout* por parte do *Smartphone* este vai usar o actual sistema transaccional, isto é, verificar se o *PIN* inserido no terminal multibanco corresponde ao *PIN* do cartão usado para a transacção, podendo assim, aceitar ou recusar o pagamento.

Caso o cliente não tenha este serviço activo, o sistema do *ActivoBank* vai efectuar a transacção através do método actual da *SIBS*, usando apenas o *PIN* inserido.

Este sistema deve de ser um sistema que o utilizador possa, fácil e rapidamente, alterar o seu estado. Com esse pensamento criou-se um sistema em que o utilizador – em apenas um toque – possa alterar o estado do serviço, de desligado a ligado e vice-versa, como se pode ver no diagrama abaixo:



Como está representado no diagrama a cima, pode-se ver que se o utilizador não estiver na aplicação basta inicia-la, inserir o código de desbloqueio e tocar na opção de activar/desactivar o serviço. Se o utilizador já se encontrar na aplicação basta tocar na opção. Através deste sistema, consegue-se activar/desactivar o serviço de pagamento de forma fácil e rápida.
















5. Testes Efectuados

Durante o desenvolvimento deste sistema surgiu a necessidade de realizar vários testes no sentido de melhorar e aperfeiçoar o próprio projecto e das aplicações desenvolvidas.

Inicialmente testou-se todo o tipo de comunicações entre o Terminal Multibanco e o sistema *SIBS* e quando se verificou que as aplicações estavam a comunicar e a funcionar correctamente, passou-se para a próxima de testes entre o sistema *SIBS* e o sistema do *ActivoBank* e o próprio sistema de aceitação/recusa de pagamento. Só após se testar o sistema de pagamento, sem se contar com o *Smartphone*, é que se testou a interacção entre o *Smartphone* e o *ActivoBank*.

Quando os testes descritos a cima foram todos aceites, foi possível testar todo o sistema de pagamento, com todas as aplicações e opções, e assim testar todo o sistema de pagamento.

Desta maneira criou-se uma tabela que foi preenchida com as funcionalidades pretendidas e os seus intervenientes:

	Cliente	<i>Smartphone</i>	<i>ActivoBank</i>	<i>SIBS</i>
Autenticação				-
Importação de Dados	-			-
Actualização de Dados	-		-	-
Pedido de Pagamento	-			
Confirmação/Recusa de Pagamento				
Activar/Desactivar Serviço	-			-

Legenda:

Sucesso - 

Insucesso - 

Incompleto - 

Contudo, a meio do desenvolvimento das aplicações, devido a um pequeno limitador que o simulador *Android* para o *Eclipse IDE* ao nível de servidores *Socket* tem, que não permite a recepção de novas conexões vindas da rede, foi quase impossível fazer o *debug* completo da aplicação. Desta forma todas as alterações efectuadas tiveram de ser testadas directamente no *Smartphone*, dificultando assim a percepção e captação de possíveis erros. Assim tentou-se, a cada utilização da aplicação no *Smartphone*, que todas as funções – possíveis de serem feitos testes – conseguissem efectuar as funcionalidades pretendidas.

6. Conclusão

No decorrer deste projecto foram aplicadas várias matérias leccionadas no decorrer da Licenciatura desde a Modelação até ao Desenvolvimento das aplicações.

As aplicações JAVA – Terminal Multibanco, *SIBS*, Sistema *ActivoBank* – foram pensadas para serem um sistema consola, pois não fazia sentido ter um interface gráfico, já a aplicação do *Smartphone* foi pensada de forma a que todos os ecrãs, excepto o de confirmação de pagamento, possam ser substituídas pelos ecrãs da aplicação já existente.

Os resultados obtidos foram bastante satisfatórios, onde se espera que numa futura implementação deste problema a solução estudada possa servir de referência.

7. Bibliografia e Referencias

- Slides das Aulas de Analise e Concepção de Sistemas
- Exemplos e exercícios efectuados nas aulas de Redes de Computadores
- <http://developer.android.com>
- <http://docs.oracle.com/javase/1.5.0/docs/api/>

8. Anexos

Junto a este relatório segue-se os Diagramas de Classe de ambos os sistemas – *SMS* e *Smartphone* – assim como os Diagramas de Iterações dos mesmos.

Anexo 1 - Diagrama de Classes Via Smartphone.pdf

Anexo 2 - Diagrama de Classes Via SMS.pdf

Anexo 3 - Iterações Via Smartphone.pdf

Anexo 4 - Iterações Via SMS.pdf

As soluções desenvolvidas e o *driver* usado encontram-se em pastas separadas para se poder navegar no código desenvolvido.