LICENCIATURA INFORMÁTICA

PROJECTO FINAL DE CURSO

GESTÃO DE REDE FERROVIÁRIA

AUTOR: PEDRO MAURÍCIO, Nº. 2102922

ORIENTADOR: PEDRO MALTA, MESTRE

JAN/2006

ÍNDICE

1. Introdução

Pretende-se com o desenvolvimento deste projecto demonstrar os conhecimentos adquiridos ao longo do Curso de Informática.

O objectivo é a criação de um simulador de uma rede ferroviária, o mais real possível, utilizando uma arquitectura cliente-servidor integrada com uma base de dados relacional.

O projecto será implementado através do Visual Studio .Net 2003 na linguagem de programação C#, a informação deverá ser acedida através de uma base de dados relacional, utilizando-se para esse efeito o Microsoft SQL Server 2000.

1.1 Enquadramento

A arquitectura .Net FrameWork disponibiliza diversas formas de comunicação, tais como, *Web Services*, *Messaging* ou *Sockets*. A vantagem da utilização de Sockets baseia-se no facto de todas as classes de acesso à rede existentes no System.Net serem construídas tendo por base a implementação de sockets. A comunicação por Sockets será utilizada para desenvolver a comunicação via TCP/IP entre diversos equipamentos. Serão criados dois interfaces gráficos, a consola da aplicação servidor e a consola da aplicação cliente.

1.2 Arquitectura Cliente-Servidor

Este projecto utiliza duas aplicações, consola e cliente, que comunicam entre si através de *sockets*. A comunicação por *sockets* pode ser efectuada de duas formas: síncrona ou assíncrona.

A comunicação síncrona utiliza chamadas IO ao sistema de forma bloqueante, isto é, enquanto a aplicação consola está à "escuta" num determinado porto, dos dados enviados pelo cliente, a mesma fica bloqueada enquanto essa condição não se verificar, isto é, a aplicação fica parada até receber os dados. Caso outro cliente efectue uma ligação, à aplicação não poderá processar essa ligação uma vez que está bloqueada pela ligação do primeiro cliente.

Na comunicação assíncrona enquanto a aplicação servidor está a escutar ou a receber dados de um cliente, poderá ainda receber ligações de outros clientes, uma vez que utiliza chamadas IO ao sistema de forma não bloqueante. Esta funcionalidade é obtida através de uma *thread* (que funciona ao nível do Sistema Operativo) que escuta o *socket* e invoca uma função de *callback*, esta função permite que a aplicação continue a processar outras ligações, enquanto espera pelos dados.

Pelos motivos anteriormente referidos, e por se tratar de uma aplicação que interage com outras aplicações (clientes), optou-se por utilizar a comunicação assíncrona nas chamadas IO ao *Socket*.

1.2.1 Cliente

O comboio deverá aceder à rede exclusivamente através da consola (servidor), efectuado e recebendo pedidos/ordens, assim sendo, o comboio deverá funcionar automaticamente do seguinte modo:

- Efectuar a ligação à consola;
- Pedir / Receber ID de rede;
- Pedir / Receber percurso;
- Receber ordens (acelera, parar e abrandar);
- Emitir periodicamente a sua localização;
- Efectuar registos (log's) de toda a sua actividade;

Deverá ser possível ter mais que um comboio em circulação, comboios esses que poderão ser de empresas diferentes, com horários e acessos concorrentes a plataformas das estações. Os comboios deverão circular na rede através de equações físicas de movimento, calculando-se dessa forma suas velocidades, podendo-se efectuar alterações à sua velocidade.

1.2.2 Servidor

A aplicação-servidor ou consola será o "mecanismo" de gestão da rede, onde será visível o diagrama da rede (linhas, estações e semáforos), o estado e propriedades de cada elemento da rede. A consola deverá funcionar em simultâneo de duas formas.

Primeiro deve comportar-se como uma aplicação-servidor e processar automaticamente aos pedidos dos clientes (comboios), assim sendo, a consola deverá funcionar automaticamente do seguinte modo:

- Gerir a comunicação de vários clientes em simultâneo;
- Atribuir ***ID*** aos clientes;
- Enviar diagrama da rede aos clientes;
- Enviar percurso aos clientes
- Efectuar registos de todas as suas actividades;

Segundo, deverá permitir que o Gestor da rede faça o registo de material circulante (comboios) e estrutura física da rede (linhas e estações) na base de dados, registo de horários, activar/desactivar semáforos e dar ordens de paragem ou início de marcha aos comboios.

1.3. Comunicação por Sockets

Como já foi referido anteriormente este projecto utiliza duas aplicações, consola e cliente, que comunicam entre si através de *sockets*. Cada aplicação cliente tem o seu próprio *socket* que comunica com o *socket* da aplicação servidor.

Sempre que é efectuada uma nova ligação é criado um *socket* (m_socket) temporário para assegurar a comunicação de dados entre a aplicação servidor e a aplicação cliente, libertando assim o *socket* principal da aplicação servidor, conforme o seguinte esquema.

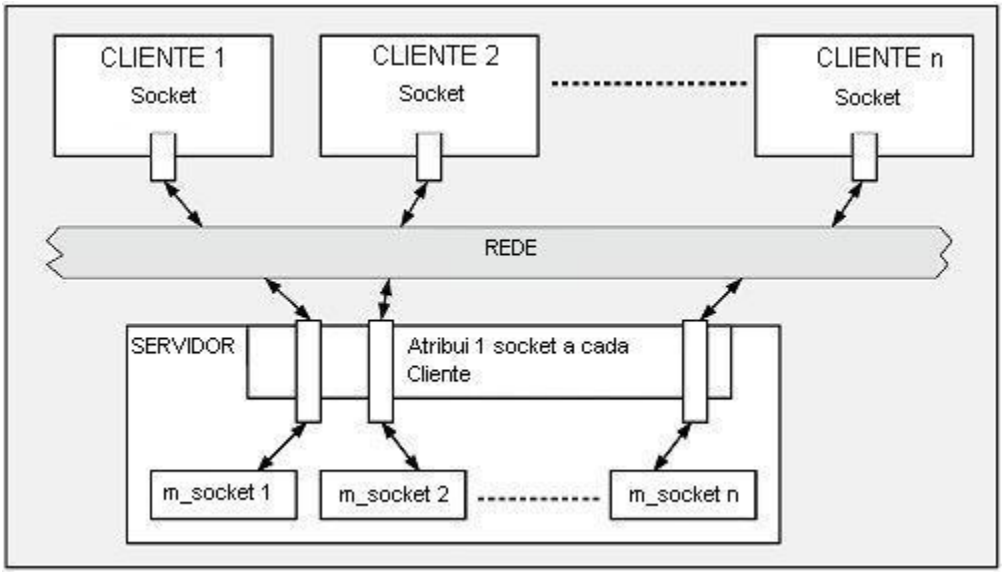


Figura 1 – Arquitectura da comunicação por sockets

2. Análise da Rede Ferroviária

A estrutura física da rede ferroviária é composta essencialmente por linhas e estações. As linhas podem ser de três tipos: normais, as linhas de plataforma existentes nas estações e as linhas dos parques. As linhas terão necessariamente uma sinalização. No desenvolvimento deste projecto optou-se por criar uma estrutura semelhante à existente na Linha do Sul, que corresponde ao percurso entre as estações de Setúbal e Roma-Areeiro.

2.2 Estações

O critério de classificação das Estações é resumidamente abaixo apresentado:

Tabela 1 – Tipologia das Estações

TIPOLOGIA	DESCRIÇÃO
A	Estações com características de terminal, de grande dimensão, com concentração de vários serviços, bem como das adequadas instalações de apoio ao passageiro. Apresentam um elevado fluxo de passageiros.
B	Estações essencialmente suburbanas, recentemente remodeladas, de grande dimensão, com adequadas instalações de apoio ao passageiro. Apresentam um elevado fluxo de passageiros.
C	Estações de média dimensão, a maioria das quais remodeladas. Apresentam um significativo fluxo de passageiros.
D	Estações e apeadeiros, geralmente dotados de Edifício de Passageiros. O fluxo de passageiros é limitado.
E	Estações e apeadeiros de pequenas dimensões com reduzido tráfego de passageiros.

De acordo com a tipologia indicada, as estações da Base de Dados, terão a seguinte classificação:

Tabela 2 – Classificação das Estações

ESTAÇÃO	CLASSIFICAÇÃO
Setúbal	C
Palmela	D
Venda do Alcaide	D
Pinhal Novo	C
Penalva	D
Coina	C
Fogueteiro	C
Foros da Amora	C
Corroios	C
Pragal	B
Campolide – A	C
Sete Rios	B
Entrecampos	B
Roma-Areeiro	C

As Estações terão várias linhas de circulação e prestarão os serviços a seguir descriminados:

Tabela 3 – Classificação das Estações

ESTAÇÕES	Nº LINHAS	SERVIÇOS ADICIONAIS		SERVIÇOS AUXILIARES			
		MANOBRAS	PARQUE	LIMPEZA	ABS. AGUA	INF. PÚBLICO	ABS. GASOLEO
Setúbal	IV	S	S	S	S	S	S
Palmela	IV	N	N	N	N	S	N
Vda do Alcaide	II	N	N	N	N	S	N
Pinhal Novo	VI	S	N	N	N	S	N
Penalva	III	S	S	N	S	S	S
Coina	IV	N	N	S	N	S	N
Fogueteiro	IV	N	N	N	N	S	N
Foros da Amora	II	N	N	N	N	S	N
Corroios	II	N	N	N	N	S	N
Pragal	IV	N	N	N	N	S	N
Campolide – A	VII	N	N	N	N	S	N
Sete Rios	IV	N	N	N	N	S	N
Entrecampos	VIII	N	N	N	N	S	N
Roma-Areeiro	IV	S	S	S	N	S	N

2.3 Linhas

A circulação dos comboios será efectuada através de um sistema de linha dupla, isto é, existirá uma circulação em dois sentidos entre estações. Como já foi anteriormente mencionado existem três tipos de linha: normais, plataforma e parque. As linhas deverão necessariamente conter sinalização, a qual será diversa, no entanto será obrigatório a utilização de sinais luminosos que servirão de acesso ou recusa à utilização de cada segmento de linha, isto é, cada segmento de linha terá um sinal (entre outros) no seu início, o qual indicará a recusa ou permissão para a ceder a esse segmento de linha.

A circulação nas linhas poderá ainda ser condicionada devido a obras, o que fará com que a velocidade de circulação permitida em cada segmento de linha seja reduzida para metade.

Em relação ao estado das linhas, as mesmas serão classificadas conforme o seguinte quadro discriminativo:

Tabela 3 – Classificação das Linhas

ESTADOS	
0	Livre, Sinal Aberto
1	Ocupada
2	Livre, Sinal Fechado
3	Livre, Condicionada

2.4 Comboios

No que diz respeito ao material circulante, o mesmo será classificado através dos serviços prestados, conforme o seguinte quadro discriminativo:

Tabela 4 – Classificação dos Comboios

SERVIÇO	DESCRIÇÃO
SUB1	Serviços suburbanos de passageiros com uma frequência igual ou superior a seis comboios por hora, nas horas de ponta.
SUB2	Serviços suburbanos de passageiros com uma frequência inferior a seis comboios por hora, nas horas de ponta.
IC	Serviços nacionais regulares de alta qualidade, intercidades e internacionais de passageiros.
OSP	Outros serviços de passageiros de médio e longo curso.
MI	Serviços de mercadorias internacionais ou tipo expresso.
MN	Serviços de mercadorias nacionais.
MN	Marchas em vazio.
OUTROS	Outros serviços, nomeadamente marchas de ensaio e para formação de pessoal ou comboios de empreiteiros.

Em relação ao estado do material circulante, o mesmo será classificado conforme o seguinte quadro discriminativo:

Tabela 5 – Classificação dos Estado dos Comboios

ESTADOS	
0	Desactivado
1	Activado, parado
2	Activado, acelerando
3	Activado, desacelerando

3. Requisitos Funcionais

De seguida serão demonstrados os principais requisitos funcionais do sistema, apresentados sob a forma de casos de utilização, diagrama de sequência e diagrama de classes.

3.1. Principais Casos de Utilização

Nos casos de utilização, teremos o caso de utilização principal, que demonstra os aspectos funcionais ou orientadores do projecto e os seus subsequentes casos de utilização (infra-estrutura, material circulante, horários e cliente-servidor).

3.1.1. Caso de Utilização Global

O caso de Utilização Global demonstra as principais funcionalidades existentes na aplicação consola, funcionalidades essas, exercidas pelo operador/administrador da aplicação. As principais funcionalidades são: a gestão da infra-estrutura (linhas e estações); a gestão do material circulante (comboios); a gestão de horários; a gestão da comunicação entre a consola e os clientes.

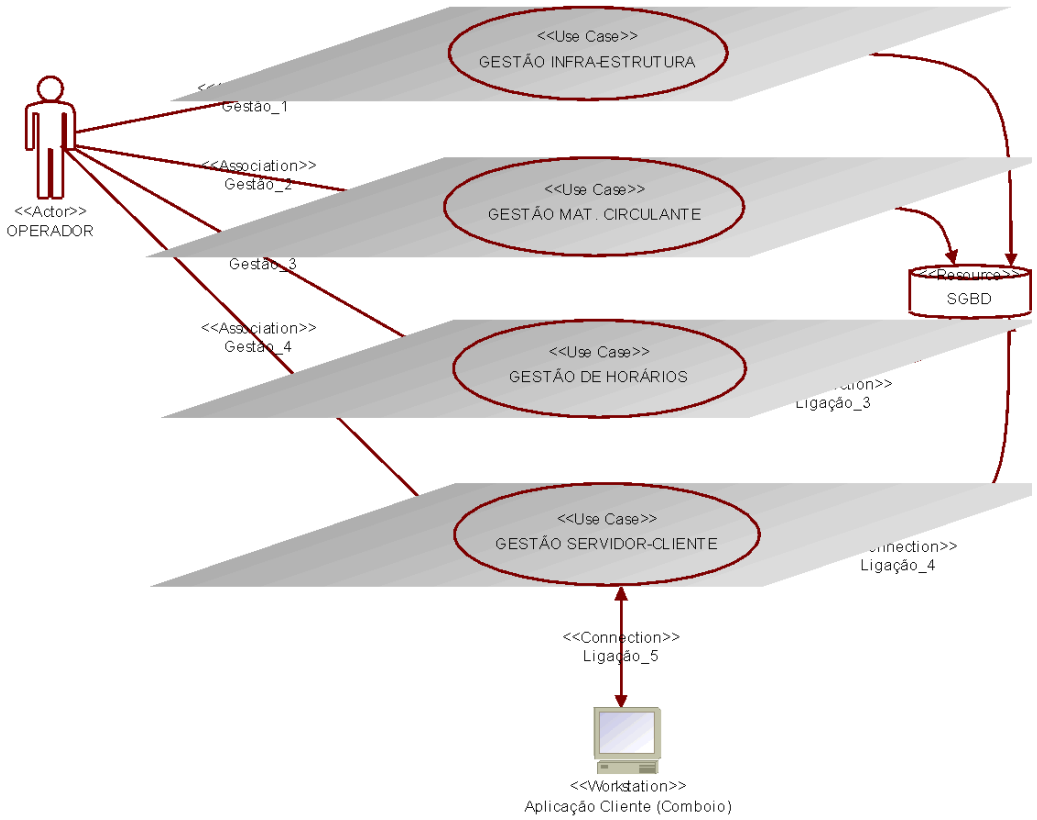


Figura 2 –Visão geral do sistema (diagrama de casos de utilização simplificado)

3.1.2. Casos de Utilização Infra-Estrutura

O caso de Utilização Infra-Estrutura demonstra as principais funcionalidades existentes na aplicação consola, funcionalidades essas, exercidas pelo operador/administrador da aplicação.

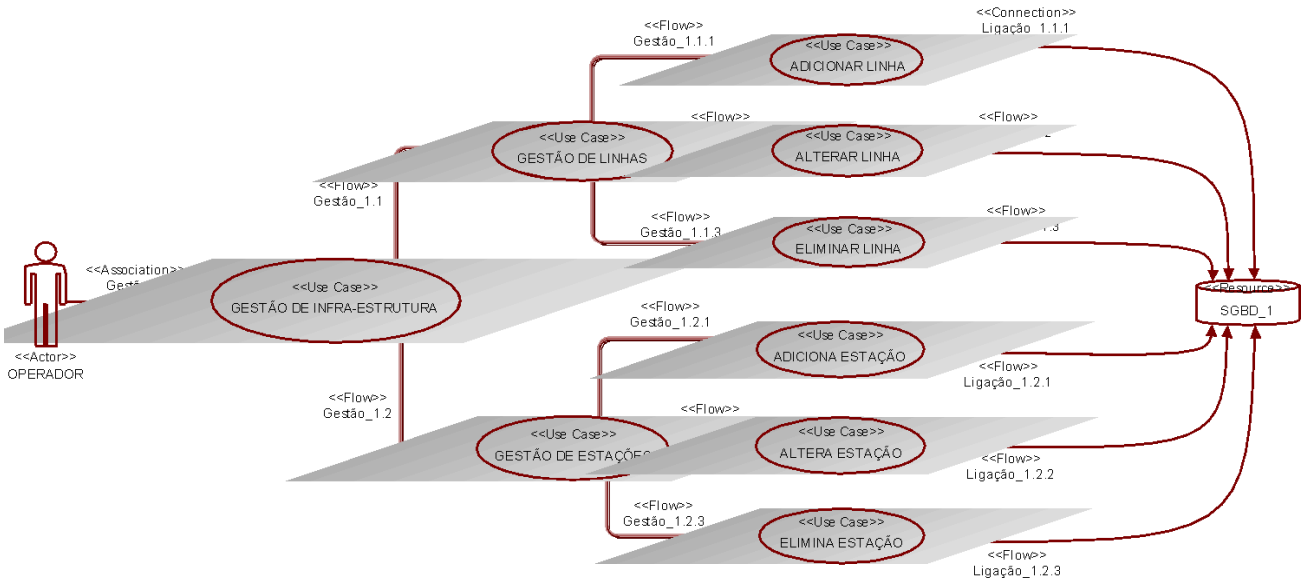


Figura 3 – Caso de Utilização Infra-Estrutura

3.1.3. Caso de Utilização Material Circulante

O caso de utilização Material Circulante demonstra as principais funcionalidades existentes na aplicação consola, funcionalidades essas, exercidas pelo operador/administrador da aplicação.

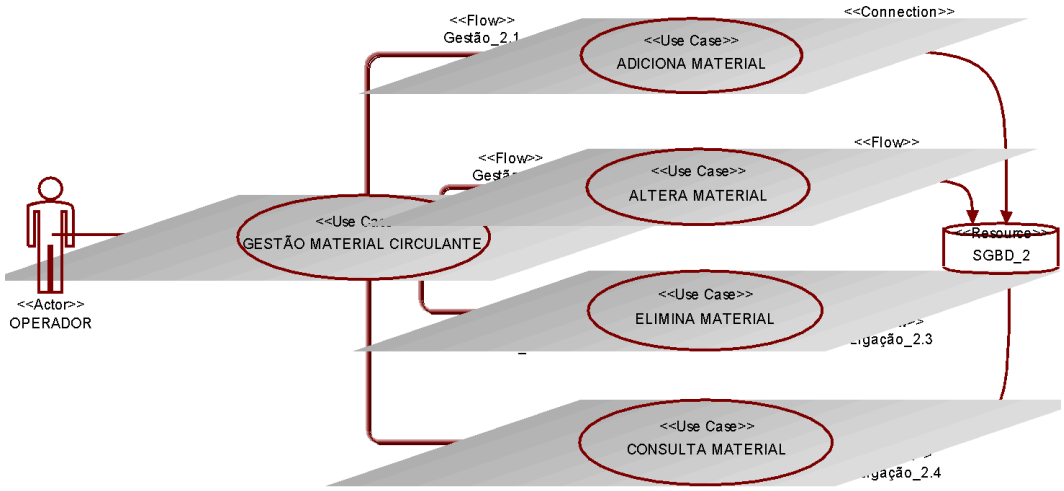


Figura 4 – Caso de Utilização Material Circulante

3.1.4. Caso de Utilização Horário

O caso de utilização Horário demonstra as principais funcionalidades existentes na aplicação consola, funcionalidades essas, exercidas pelo operador/administrador da aplicação.

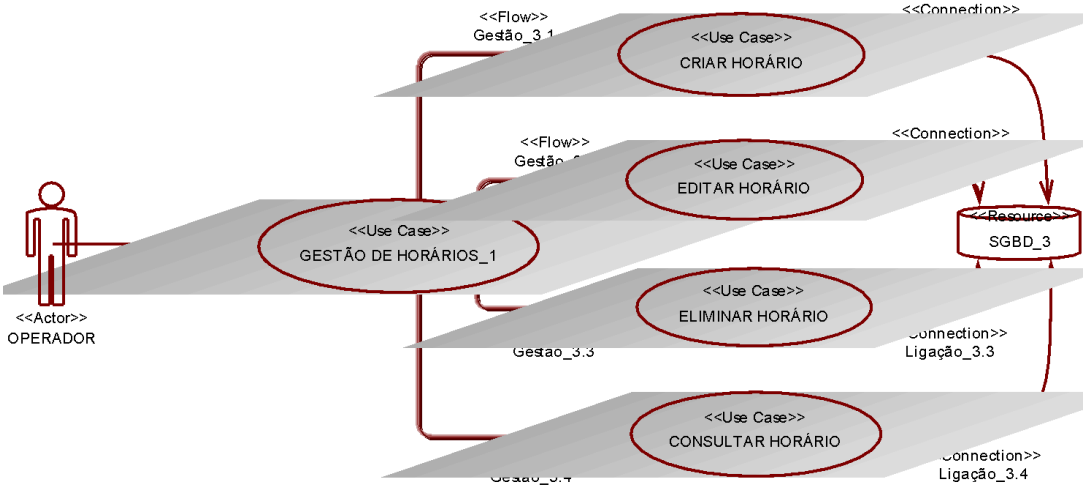


Figura 5 – Caso de Utilização Horário

3.1.4. Caso de Utilização Cliente-Servidor

O caso de utilização Cliente-Servidor demonstra as principais funcionalidades existentes na aplicação consola, funcionalidades essas, exercidas pelo operador/administrador da aplicação. Cabe ao operador/administrador iniciar/parar as ligações entre a consola e os clientes e apagar os registos das comunicações entre os mesmos.

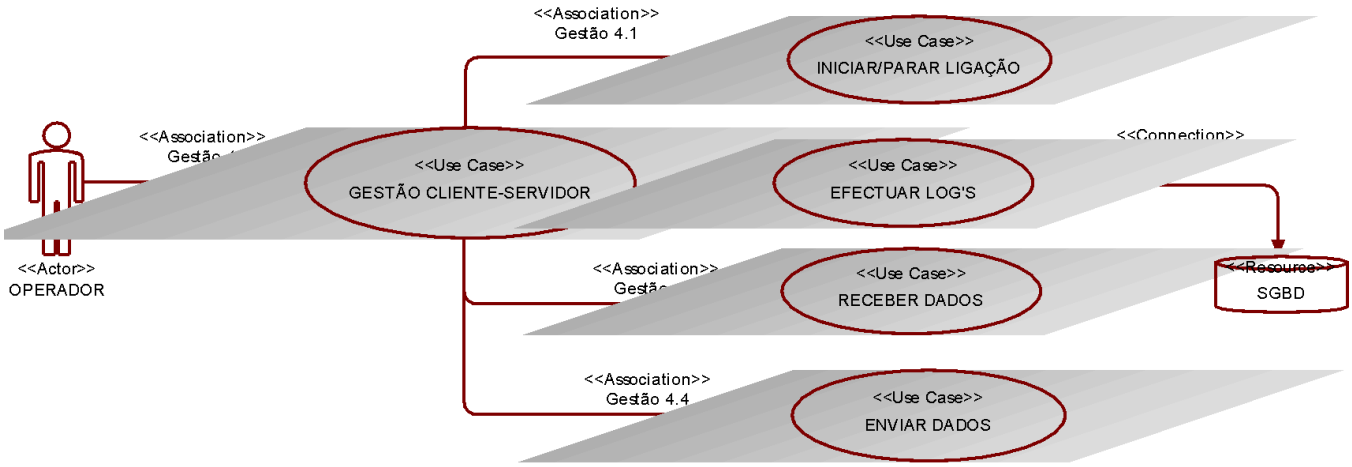


Figura 6 – Caso de Utilização Cliente-Servidor

3.2. Diagrama de Classes

O seguinte diagrama de classes baseia-se na análise da Rede Ferroviária.

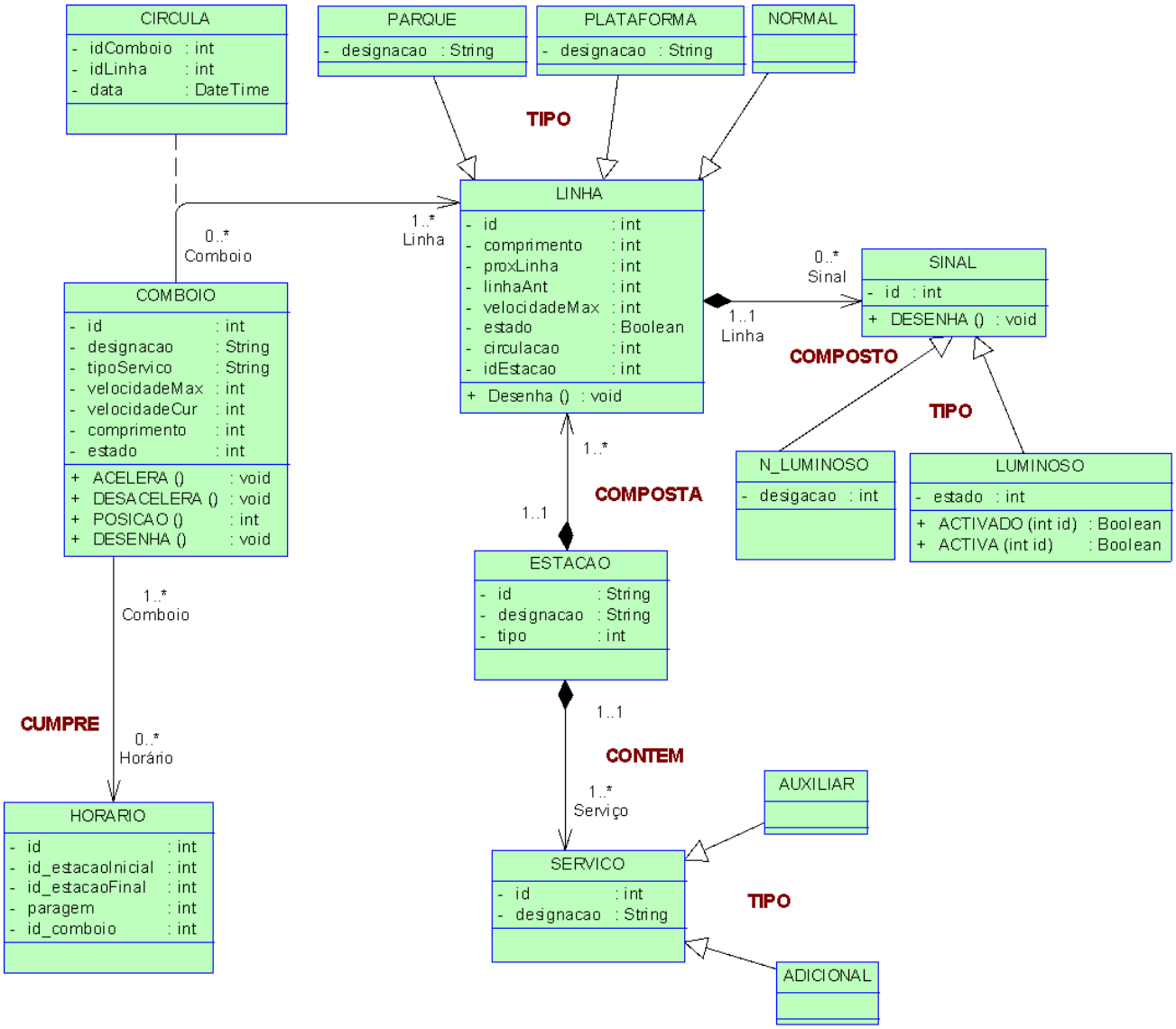
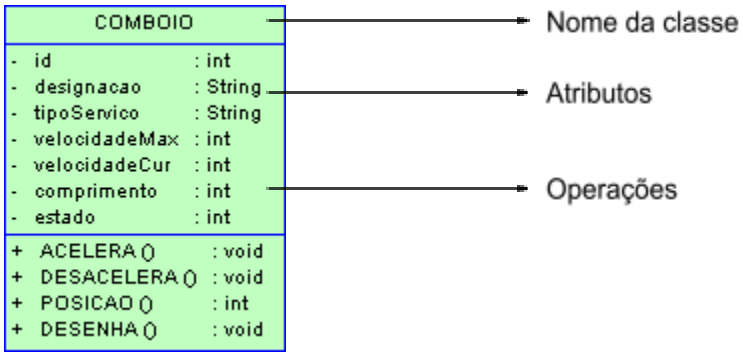


Figura 7 – Diagrama de Classes

3.2.1. Classe Comboio

A classe comboio representa no sistema os clientes (comboios) que se ligam à aplicação servidor. Esta classe é uma classe entidade, pois, para cada entidade do mundo real presente em cada caso de utilização e acerca da qual o sistema necessita de armazenar informação, cria-se uma classe entidade.

Esta classe tem a seguinte constituição:

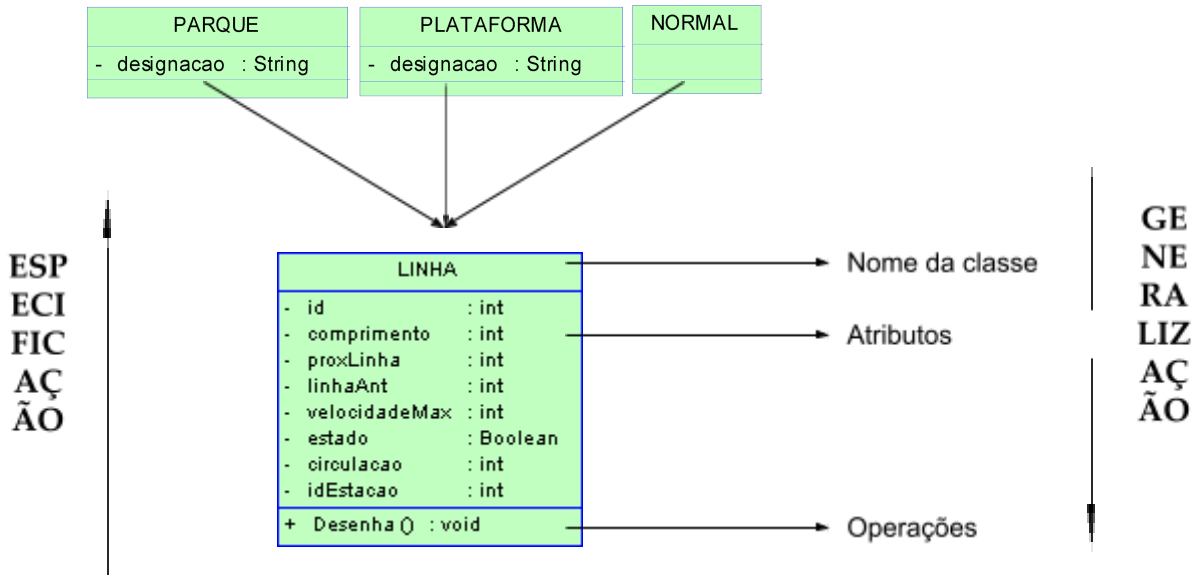


3.2.2. Classe Linha

A classe linha representa no sistema o traçado, no qual, os objectos da classe comboio se movimentam. Tal com a classe Comboio, também esta classe é uma classe entidade, pois, pois também é necessário para o sistema armazenar informação.

Esta classe é uma classe genérica, que tem como três subclasses (Parque, Plataforma, Normal), que no modelo de dados deixarão de existir, ficado apenas a classe linha, com mais um atributo (*tipo={parque, plataforma, normal}*).

Esta classe tem a seguinte constituição:



3.2.3. Classe Circula

A classe circula é uma classe relacional e entidade, relacional porque faz a relação entre duas classes (comboio e linha), entidade porque o sistema necessita armazenar informação sobre ela.

Esta classe tem a seguinte constituição:

3.2. Diagrama de Sequência

O seguinte diagrama de sequência é uma representação genérica dos diagramas de sequência para as classes Linha, Estacão, Comboio e Horário.

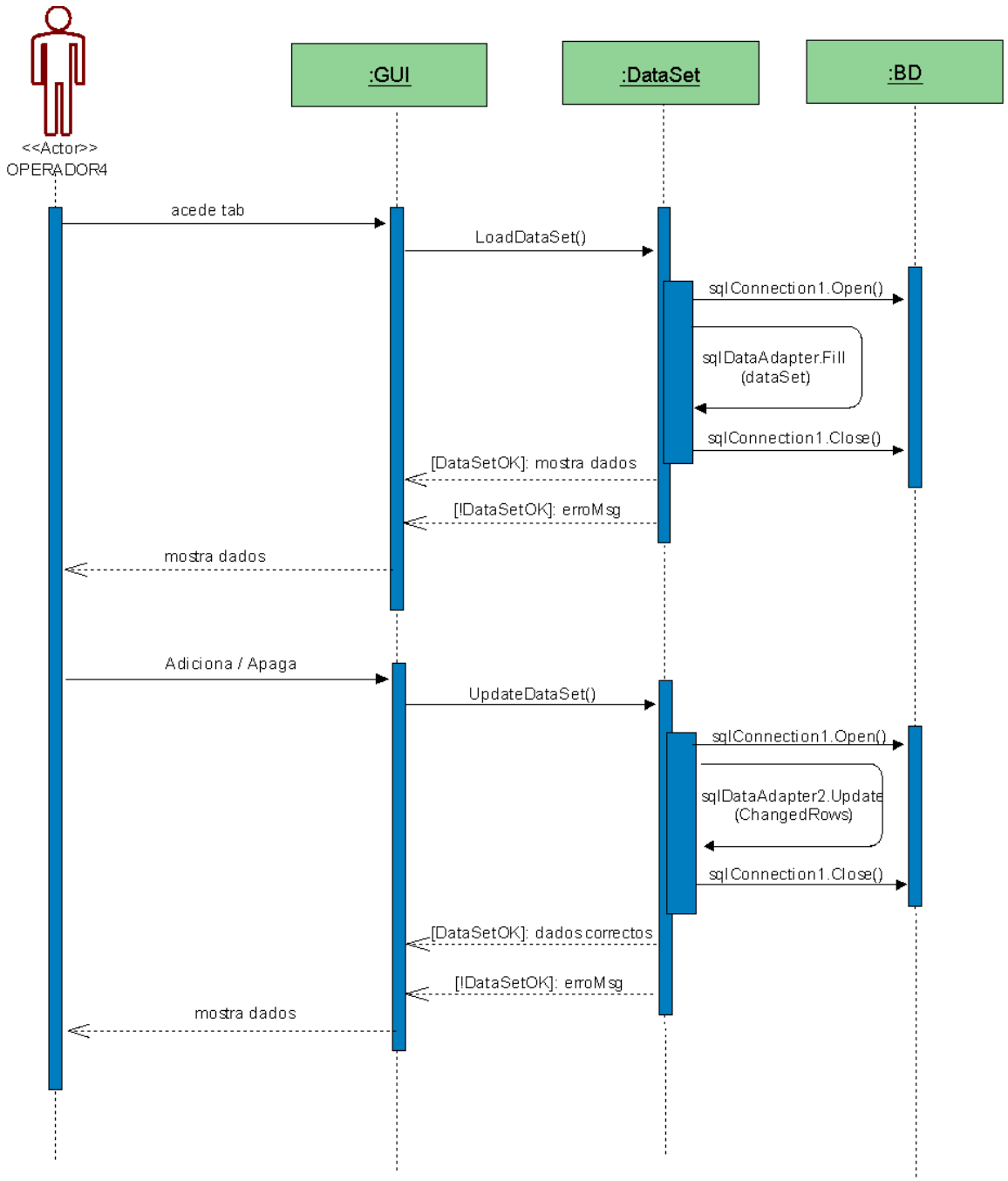


Figura 8 – Diagrama de Sequência

3.3. Modelo de Dados Optimizado

Comboio (id, designacao, tipoServico, velocidadeMax, velocidadeCur, comprimento, estado)

Linha (id, comprimento, proxLinha, linhaAnt, velocidadeMax, estado, circulacao, categoria={normal, plataforma, parque}, *idEstacao*)

Sinal (id, designacao, tipo={luminoso,nLuminoso}, estado)

Estacão (id, designacao, tipo)

Servico (id, designacao, tipo={adicional, auxiliar})

Fornece (Estacão.id, *Serviço.id*)

Horario (id, *Estacão.id*, *Linha.id*, entrada, saida)

Cumpre (Comboio.id, id, *Estacão.id*, *Linha.id*, data)

3.4.1. “Scripts” SQL

CRIAÇÃO DA TABELA COMBOIO

```
create table comboio(
    id int IDENTITY(1,1) PRIMARY KEY,
    designacao varchar(50) NOT NULL,
    tipoServico varchar(50) NOT NULL,
    velocidadeMax int NOT NULL,
    velocidadeCur int NOT NULL,
    comprimento int NOT NULL,
    estado int NOT NULL
)
```

INTRODUÇÃO EM COMBOIO

```
insert into comboio VALUES (’FERTAGUS-1’, ’SUB1’, 140, 0, 200, 0)
insert into comboio VALUES (’FERTAGUS-2’, ’SUB2’, 140, 0, 200, 0)
insert into comboio VALUES (’ALFA-PENDULAR’, ’IC’, 240, 0, 180, 0)
insert into comboio VALUES (’INTERCIDADES’, ’IC’, 150, 0, 150, 0)
insert into comboio VALUES (’REGIONAL-1’, ’OSP’, 140, 0, 200, 0)
insert into comboio VALUES (’MERC-1’, ’SNM’, 120, 0, 400, 0)
insert into comboio VALUES (’FERTAGUS-1’, ’SUB1’, 140, 0, 200, 0)
```

CRIAÇÃO DA TABELA ESTACAO

```
create table estacao (
    id int IDENTITY(1,1) PRIMARY KEY,
    designacao varchar(50) NOT NULL,
    tipo char NOT NULL,
)
```

INTRODUÇÃO EM ESTACAO

```
insert into estacao VALUES ('ESTAÇÃO DE SETÚBAL', 'C')
insert into estacao VALUES ('ESTAÇÃO DE PALMELA', 'D')
insert into estacao VALUES ('ESTAÇÃO DE VENDA DO ALCAIDE', 'D')
insert into estacao VALUES ('ESTAÇÃO DE PINHAL NOVO', 'C')
insert into estacao VALUES ('ESTAÇÃO DE PENALVA', 'D')
insert into estacao VALUES ('ESTAÇÃO DE COINA', 'C')
insert into estacao VALUES ('ESTAÇÃO DE FOGUETEIRO', 'C')
insert into estacao VALUES ('ESTAÇÃO DE FOROS DA AMORA', 'C')
insert into estacao VALUES ('ESTAÇÃO DE CORROIOS', 'C')
insert into estacao VALUES ('ESTAÇÃO DE PRAGAL', 'C')
insert into estacao VALUES ('ESTAÇÃO DE CAMPOLIDE-A', 'C')
insert into estacao VALUES ('ESTAÇÃO DE SETE RIOS', 'B')
insert into estacao VALUES ('ESTAÇÃO DE ENTRECAMPOS', 'B')
insert into estacao VALUES ('ESTAÇÃO DE ROMA-AREEIRO, 'C')
```

CRIAÇÃO DA TABELA LINHA

```
create table linha(
    id int IDENTITY(1,1) PRIMARY KEY,
    designacao varchar(50) NOT NULL,
    comprimento int NOT NULL,
    proxLinha int NOT NULL,
    linhaAnt int NOT NULL,
    velocidadeMax int NOT NULL,
    estado int NOT NULL,
    circulacao int NOT NULL,
    categoria varchar(50) NOT NULL,
    idEstacao int
```

```
)
```

INTRODUÇÃO DA TABELA LINHA DE PLATAFORMAS DE PASSAGEIROS

```
insert into linha VALUES ('LINHA 1', '300', 0, 0, 100, 0, 0, 'PLATAFORMA', 1)
insert into linha VALUES ('LINHA 2', '300', 0, 0, 100, 0, 0, 'PLATAFORMA', 1)
insert into linha VALUES ('LINHA 3', '300', 0, 0, 100, 0, 0, 'PLATAFORMA', 1)
insert into linha VALUES ('LINHA 4', '300', 0, 0, 100, 0, 0, 'PLATAFORMA', 1)
```

CRIAÇÃO DA TABELA SINAL

```
Create table sinal (
    id int NOT NULL,
    idLinha int NOT NULL,
    designacao varchar(50) NOT NULL,
    tipo varchar(50) NOT NULL,
    estado int NOT NULL
    CONSTRAINT [PK_sinal] PRIMARY KEY CLUSTERED
    (
        id,
        idLinha
    )  ON [PRIMARY]
) ON [PRIMARY]
```

INTRODUÇÃO DA TABELA SINAL REFERENTES ÀS PLATAFORMAS DA LINHA DE SETÚBAL

```
Insert into sinal VALUES (1, 1,'SEMAFORO', 'LUMINOSO', 0)
Insert into sinal VALUES (2, 2,'SEMAFORO', 'LUMINOSO', 0)
Insert into sinal VALUES (3, 3,'SEMAFORO', 'LUMINOSO', 0)
Insert into sinal VALUES (4, 4,'SEMAFORO', 'LUMINOSO', 0)
```

CRIAÇÃO DA TABELA HORARIO

```
create table horario(
    id int NOT NULL,
    idEstacao int NOT NULL,
    designacao varchar(50) NOT NULL,
    tipo varchar(50) NOT NULL
    CONSTRAINT [PK_servico] PRIMARY KEY CLUSTERED
    (
        id,
        idEstacao
    )  ON [PRIMARY]
) ON [PRIMARY]
```

--INTRODUÇÃO EM SERVIÇO PARA AS ESTAÇÕES DE SETÚBAL E PINHAL NOVO--

```
insert into servico VALUES (1, 1, 'MANOBRAS', 'ADICIONAL')
insert into servico VALUES (2, 1, 'PARQUE', ' ADICIONAL')
insert into servico VALUES (3, 1, 'LIMPEZA', 'AUXILIAR')
insert into servico VALUES (4, 1, 'INFORMAÇÃO PÚBLICO', 'AUXILIAR')
insert into servico VALUES (5, 4, 'MANOBRAS', 'ADICIONAL')
insert into servico VALUES (6, 4, 'INFORMAÇÃO PÚBLICO', 'AUXILIAR')
```

3.4.2. Diagrama de Relações

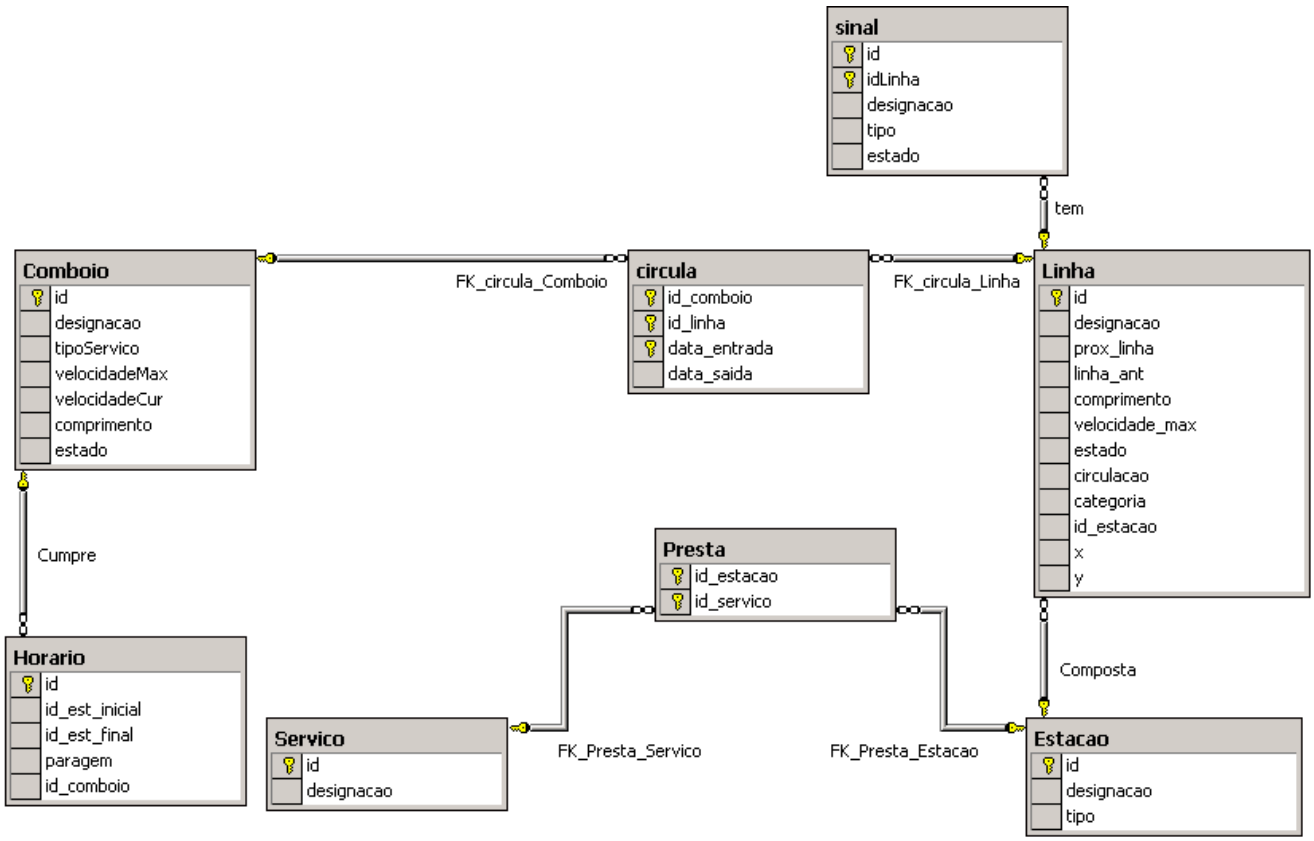


Figura 9 – Diagrama de Relações da Base de Dados

4. Arquitectura do Sistema

A implementação física do projecto baseia-se na comunicação entre a aplicação cliente a correr em diversos computadores pessoais (pc's) e a aplicação servidor a correr no servidor aplicacional, que serve também de repositório de dados, através do Sistema de Base de Dados SQL.

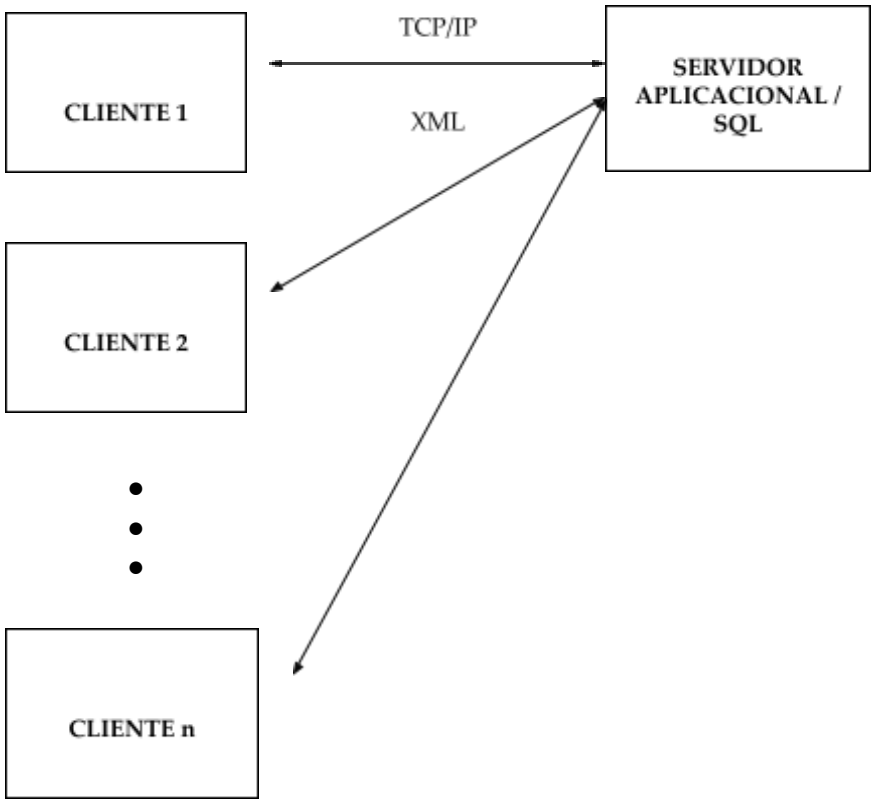
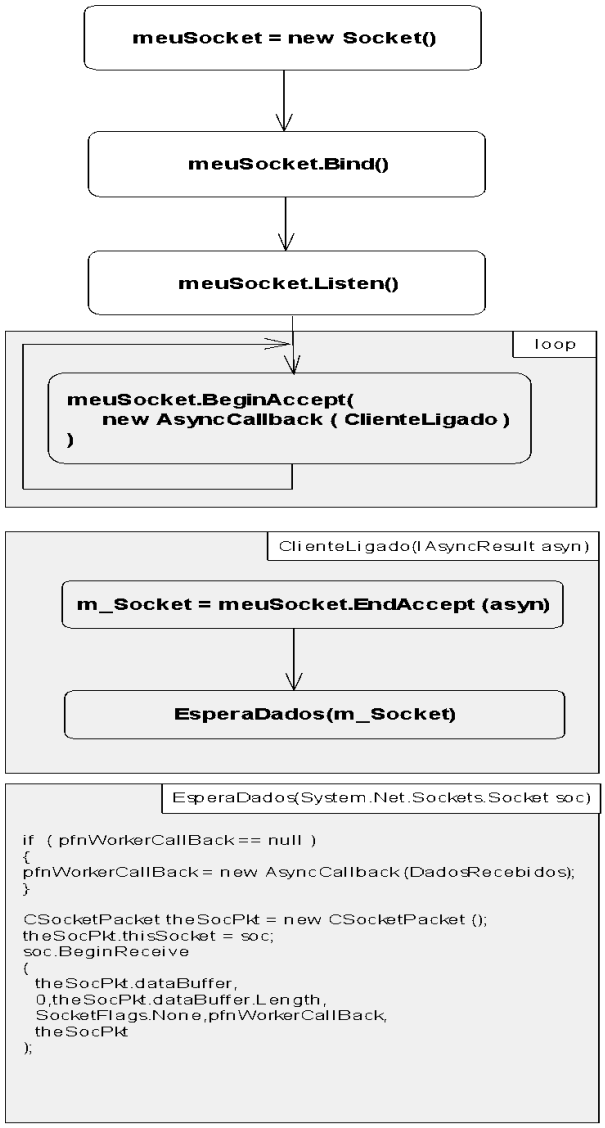


Figura 10 – Diagrama Físico

6. Implementação

Como foi referido no início deste documento, o desenvolvimento do código será efectuado na linguagem de programação Visual C#. Esta linguagem de programação é orientada a objectos, sendo a sua sintaxe bastante similar com Java, linguagem esta (Java) leccionada nas cadeiras do Curso de Informática da Universidade.

6.1. Implementação do *Socket* Servidor



1. É criado o socket que vai gerir a comunicação entre a aplicação servidor e os clientes;

2. É efectuado através do método ***Bind***, a associação entre o *socket* e *localEndPoint* (endereço IP e porta);

3. O *socket* é colocado em estado de escuta;

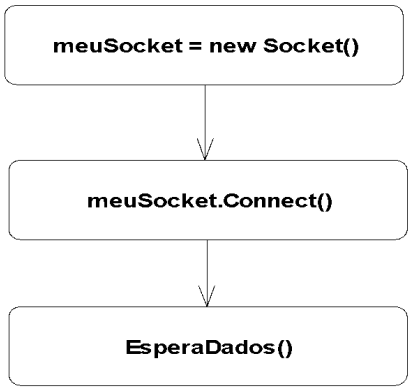
4. Inicia a operação assíncrona para receber ligações dos clientes. Quando existir uma ligação o método *AsyncCallback* retornará a rotina ***ClienteLigado***.

5. A rotina ***ClienteLigado*** cria o socket m_Socket, passando assim, a gestão da comunicação para o mesmo. É chamada a rotina ***EsperaDados***, que recebe como parâmetro o socket criado.

6. A rotina ***EsperaDados*** faz a gestão dos dados recebidos através do método assíncrono ***BeginReceive***.

Figura 11 – Esquema da implementação do socket da aplicação servidor

6.2. Implementação do *Socket* Cliente



1. É criado o socket que vai efectuar a comunicação com a aplicação servidor;

2. É efectuado através do método ***Connect***, a ligação com o socket da aplicação servidor;

3. A rotina ***EsperaDados*** faz a gestão dos dados recebidos através do método assíncrono ***BeginReceive***.

```
EsperaDados(System.Net.Sockets.Socket soc)

if ( m_pfnCallBack == null )
{
 m_pfnCallBack = new AsyncCallback (RecebendoDados);
}

SocketPacket theSocPkt = new SocketPacket ();
theSocPkt.thisSocket = meuSocket;
m_result = meuSocket.BeginReceive
(theSocPkt.dataBuffer,
    0, theSocPkt.dataBuffer.Length,
    SocketFlags.None,
    m_pfnCallBack,
theSocPkt);
```

Figura 12 – Esquema da implementação do socket da aplicação cliente

6.3. Interface da Aplicação Servidor

A aplicação servidor permite ao operador iniciar/parar a comunicação com os clientes, através do separador Configuração, adicionar, alterar ou eliminar os dados referentes a estações, linhas, material circulante ou horários e visualizar os estado das linhas, através dos restantes separadores.

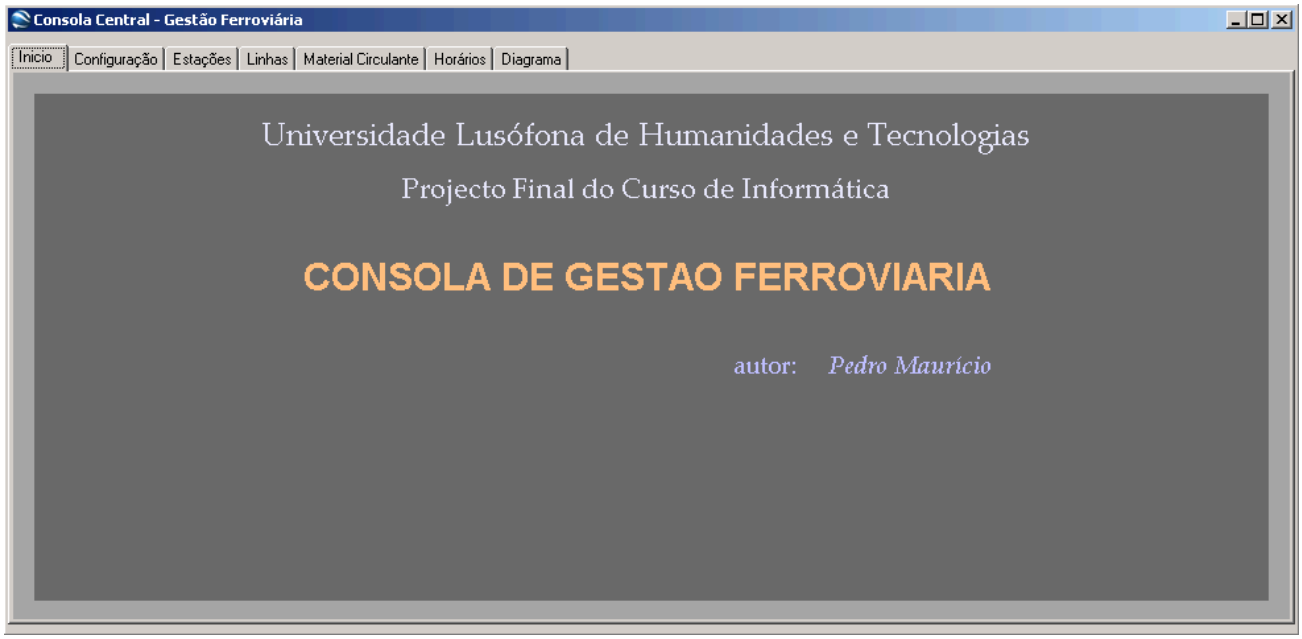


Figura 13 – Interface da Aplicação Servidor

Para cada separador, estações, linhas, material circulante e horários é efectuada uma ligação à base de dados através da classe **`System.Data.SqlClient.SqlConnection`**, sendo os dados extraídos através da classe **`sqlDataAdapter`** e colocados num **`DataSet`**, o qual se pode adicionar, alterar ou apagar dados e posteriormente actualizar a base de dados.

6.4. Interface do Cliente

A aplicação cliente permite ao operador iniciar/parar a comunicação com o a aplicação servidora, através do separador Ligação, enviar e receber pedidos (id, diagrama da rede, ordens).

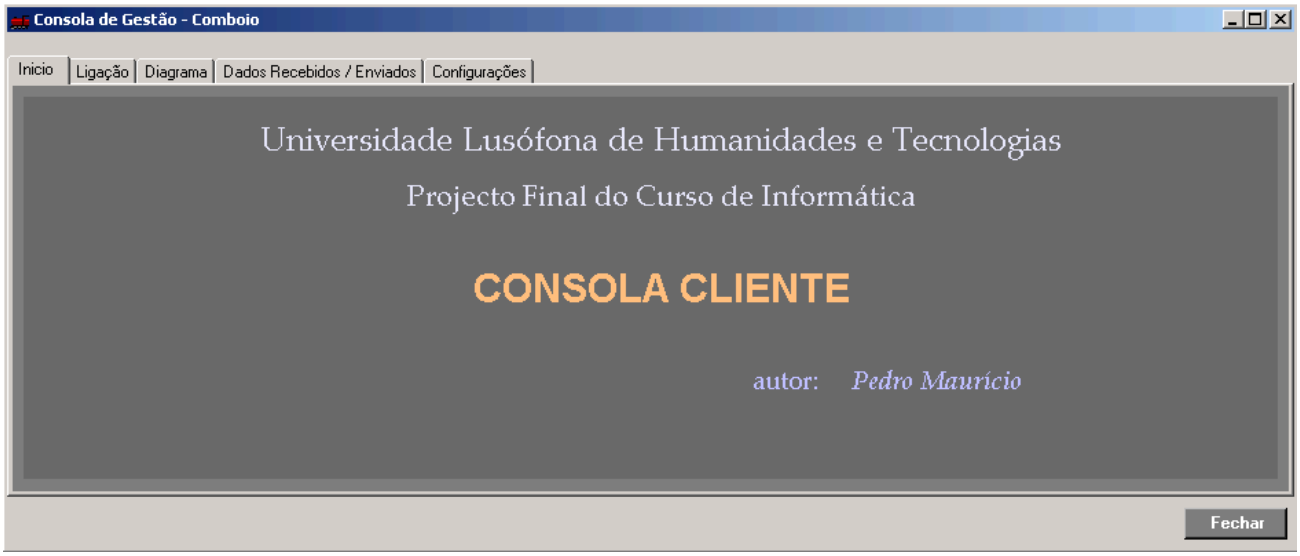


Figura 14 – Interface da Aplicação Cliente

7. Base de Dados

A base de dados utilizada no projecto é uma base de dados relacional implementada em SQL 2000 Server. Tal como está referido no ponto 4 (Arquitectura do Sistema) é necessário um computador que seja ao mesmo tempo Servidor da Base de Dados e Aplicacional, isto é, a aplicação servidor irá também correr nesse equipamento.

7.1. Instalação da Base de Dados

É necessário criar no Servidor SQL a base de dados "Projecto" e posteriormente fazer a importação (restore) do ficheiro de backup existente no disco compacto (CD) em anexo.

8. Conclusão

A realização deste projecto permitiu aprofundar os conhecimentos adquiridos ao logo do curso, nomeadamente no planeamento e implementação de aplicações.

Permitiu ainda adquirir um conhecimento mais extenso sobre aplicações cliente-servidor, mais especificamente sobre as potencialidades inerentes às aplicações baseadas em Sockets, tendo-se alcançado os objectos inicialmente traçados no início deste documento.

Apesar das dificuldades inerentes ao desenvolvimento de aplicações em linguagens de programação que não eram do conhecimento do autor, depois de terminado este projecto, pode-se dizer que foi uma experiência enriquecedora, que deu a conhecer as potencialidades do Visual Studio .Net e nomeadamente o Visual C#.

Salienta-se ainda o facto de este projecto poder ser melhorado, ficando neste momento uma boa base de trabalho para aplicações de arquitectura cliente-servidor baseadas em sockets.

9. Bibliografia

Mauro Nunes e Henrique O'Neil, Fundamental de UML, FCA, 2004

Jason Price, C# Database Programming, Sybex, 2003

MSDN for Visual Studio .NET 2003

10. Anexos

Em anexo junta-se Disco Compacto (CD), com os seguintes ficheiros:

- Relatório.doc;
- ConsolaCentral.exe;
- ConsolaCliente.exe;
- Directório "BD_SQL_Projecto" com a base de dados.