MN Horários

Noémio Dória Nº 6487

Instituto Politécnico da Guarda

Escola Superior de Tecnologia e Gestão

Departamento de Informática

Dezembro de 2008

MN Horários

Noémio de Jesus da Encarnação Dória Nº 6487

RELATÓRIO SUBMETIDO COMO REQUISITO PARCIAL
PARA OBTENÇÃO DO GRAU DE LICENCIADO EM ENGENHARIA INFORMÁTICA

ORIENTADO POR PROFESSOR JOSÉ FONSECA

Instituto Politécnico da Guarda
Escola Superior de Tecnologia e Gestão
Departamento de Informática
Dezembro de 2008

À minha namorada Ana Rosa Alegria da Silva por todo o seu apoio

Resumo

O presente relatório relata a criação de uma aplicação que permitiria a criação de horários para as aulas referentes aos cursos leccionados na Escola Superior de Tecnologia e Gestão.

Para além deste objectivo, a aplicação deve estar estruturada tendo em conta a integração de duas bases de dados idênticas criadas em gestores de bases de dados diferentes. Neste, caso uma base de dados em T-SQL criada na aplicação Microsoft SQL Server 2005 e uma outra em PL / SQL criada em Oracle Server 10G R2 da *Oracle*.

A aplicação foi desenvolvida para funcionar num ambiente WEB, sendo inicialmente posto a funcionar em ambiente Microsoft Windows.

Esta foi estruturada em diferentes camadas sendo duas delas criadas em *NetBeans* da *Sun* e uma outra criada nos gestores de bases de dados.

Palavras- Chave: Integração entre bases de dados, Aplicações WEB, Modelo MVC.

Abstract

The following report describes the creation of an application that permits to create the timetables referring the courses lectured at Escola Superior de Tecnologia e Gestão.
Besides this objective the application is structured to use two identical databases created on different database management systems.
Is this case the databases will be created in T-SQL created using Microsoft SQL Server 2005 and another in PL / SQL created on Oracle Server 10G R2 da *Oracle*.
This project was developed as a Web Application being initially developed on Windows.
The Model-View-Controller arquitecture was used to develop this project using *NetBeans* for View and Controller layers and the Model layer was developed in the databases.

Keywords: Database Integration, Web Applications, MVC Model

Agradecimentos

Gostaria de agradecer inicialmente ao meu amigo e colega Filipe Caetano cujo sentido pratico ajudou corrigir alguns bugs na aplicação criada.

Agradeço ao Professor José Fonseca pela oportunidade para o desenvolvimento deste projecto e claro está pela paciência por ele demonstrada.

À minha família pelo apoio incondicional

O último, mas o maior agradecimento, vai para a minha namorada Ana Silva cujo amor e carinho foram motivação e força para ultrapassar obstáculos.

Indice

Índice de Figuras

Índice de Tabelas

1. Introdução

Este relatório tem como objectivo retratar os passos, as metodologias tomadas para o desenvolvimento do projecto do 2º semestre do 5º ano do curso de Engenharia Informática 2º ciclo, para a conclusão da Licenciatura.

O projecto proposto pelo Professor José Fonseca, visa a integração de bases de dados idênticas criadas em Sistemas Gerais de Bases de Dados diferentes. A segunda parte visa o desenvolvimento de uma aplicação para a criação de horários inicialmente pensada para a sua utilização na Escola Superior de Tecnologia e Gestão como ferramenta de apoio. Mais tarde, decidiu-se pela criação de uma aplicação Web optando pelo desenvolvimento por camadas.

Esta aplicação tem uma função

A criação de horários é uma actividade semestral que engloba uma enorme paciência, responsabilidade e claro está uma enorme complexidade. Este facto deve-se, aos números factores associados:

- O número de professores, juntamente com as suas indisponibilidades.
- O número de disciplinas e dos tipos de aulas.
- À disposição das salas.
- Às mudanças associadas à criação de um novo ano lectivo.

Tendo em conta estes factores o presente projecto visa o desenvolvimento de uma aplicação Web, através da programação em camadas. Esta aplicação, permitirá, a gestão de bases de dados idênticas criadas diferentes que são neste caso, o Oracle 10g R2 e o SQL Server 2005 da Microsoft.

Esta aplicação Web é desenvolvida nos âmbitos de diferentes áreas, tais como Engenharia de Software, Sistemas Distribuídos e Paradigmas da Programação.

2. Tecnologias

2.1. UML

A *Unified Modeling Language* (UML) é uma linguagem gráfica padrão para a elaboração da estrutura de projectos complexos de software. Pode ser empregada para visualizar, especificar, construir e documentar software.

Em 1997, a versão a UML v1.1 foi adoptada pela *Object Management Group* (OMG) (Nogueira 2005) e desde então tornou-se o padrão para a modelação de software através da unificação da linguagem de modelagem de objectos de 3 métodos líderes do mercado da altura: *Booch*, *Object Modeling Technique* (OMT) e *Objected-Oriented Software Engineering* (OOSE).

A linguagem UML é uma linguagem simples, expressiva e de grande abrangência.

A arquitectura de um sistema pode ser descrita através de 5 visões interligadas. Cada visão constitui uma projecção na organização e estrutura do sistema, focando determinado aspecto desse sistema. A UML é uma linguagem muito expressiva, abrangendo todas as visões necessárias ao desenvolvimento e implantação de sistemas:

- Visão de caso de uso: foca os comportamentos de um sistema com transparência tendo em conta todos os agentes envolvidos, como por exemplo: gerentes, analistas, programadores e utilizadores finais.
- Visão de Projecto: foca a estrutura de um sistema através da definição de classes, colaborações e as interfaces do sistema.
- Visão de Processo: foca as questões de desempenho e escalabilidade do sistema.
- Visão de Implementação: foca os elementos físicos (programas, bibliotecas, banco de dados) para a montagem do sistema.
- Visão de Implantação: foca a topologia do hardware, liberação e instalação do sistema.

Um diagrama é a apresentação gráfica de um conjunto de elementos para permitir a visualização de um sistema sob diferentes perspectivas. A linguagem UML disponibiliza diagramas específicos para a modelagem visual das 5 visões:

- Diagrama de Casos de Uso: para ilustrar as interacções que envolvem a utilização do sistema.
- Diagrama de Classes: para ilustrar a estrutura lógica
- Diagrama de Objectos: para ilustrar os objectos e a interactividade entre os mesmos
- Diagrama de Estados: para ilustrar comportamentos
- Diagrama de Componentes: para ilustrar a estrutura física do software.
- Diagrama de Interacções: composto de diagrama de sequência e diagrama de colaboração. Utilizado para ilustrar comportamentos
- Diagrama de Actividades: para ilustrar o fluxo dos eventos.

Em um sentido mais amplo, um modelo é uma simplificação da realidade. A modelação visual com UML torna a arquitectura do sistema mais compreensível, permitindo a avaliação em dimensões múltiplas.

A UML permite avaliar a aderência e a qualidade da arquitectura através de iterações precoces com os utilizadores quando os defeitos podem ser corrigidos antes de comprometer o sucesso do projecto.

Utilizando uma linguagem de modelagem padrão como a UML, os diferentes membros envolvidos podem comunicar as suas decisões sem que haja ambiguidades ou diferenças de interpretação.

A modelagem visual permite que os detalhes do processo sejam expostos ou escondidos conforme a necessidade, auxiliando o desenvolvimento de projectos complexos e extensos. Além disto, a UML ajuda a manter a consistência entre a especificação e a implementação através do desenvolvimento iterativo e do planeamento de testes em cada iteração.

2.2. Modelo MVC

O modelo *Model-View-Controller* (**MVC**) ou programação e 3 camadas é hoje o mais utilizado para a produção de aplicações Web

Esta arquitectura permite aos desenvolvedores criar componentes reutilizáveis que podem interagir umas com as outras directa ou alternadamente.

Como dito anteriormente, este modelo é constituído por 3 partes:

- Model (Modelo): onde a informação é armazenada e onde as operações relacionadas se encontram. Neste caso, associadas as bases de dados e à programação PL\SQL e T-SQL.
- Controller (Controlador ou Mediador): onde são efectuadas as acções dos utilizadores e o processamento de informação que normalmente possam provocar mudanças na camada Model. Esta camada é a responsável pela comunicação entre a camada Model e a camada View servindo de mediador. Neste caso, esta camada está implementada através de classes Java.
- View (Interface, Ver): A interface que permite ao utilizador interagir com o sistema e proceder alterações que poderão ser introduzidas, ou não, através do Controller. Neste caso apresentado através de Java Server Pages (JSP).

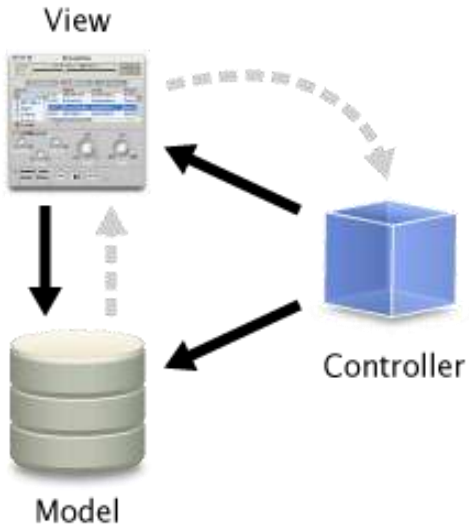

Figura 2.1-Esquema de Modelo MVC (cocoalab 2007)

2.3. Java

Em 1991, na *Sun Microsystems,* foi iniciado o *Green Project*, o berço do **Java**, uma linguagem de programação orientada a objetos. Os mentores do projecto eram Patrick Naughton, Mike Sheridan, e James Gosling. Os mesmos acreditavam que os computadores e os electrodomésticos iriam convergir de forma a se tornarem na nova tecnologia do lar.

Para provar a viabilidade desta ideia, 13 pessoas trabalharam arduamente durante 18 meses. Em 1992, em *Sand Hill Road* em *Menlo Park* foi apresentada uma versão de demonstração do *StarSeven* (*7), um controlo remoto activado através de toque, com uma interface gráfica, onde o ajudante virtual nela existente se denominava Duke o qual mais tarde se tornou na mascote hoje amplamente conhecido no mundo Java. Este controlo remoto, o *7, estava programado com uma linguagem de programação com o nome Oak, criada e baptizada por James Gosling, para que o dispositivo pudesse interagir com diversos dispositivos e aplicações.

Tendo em conta que a empresa queria pôr tal invenção a produzir resultados, rapidamente. A equipa criadora do *7 criou uma demonstração com o nome MovieWood, que iria permitir controlar equipamento multimédia tal como televisores e vídeo por demanda. Infelizmente, a empresas que disponibilizavam serviços de televisão por cabo não podiam viabilizar a produção da ideia, uma vez que não entendiam o conceito. Hoje em dia, esta é o melhor serviço que pode ser encontrado junto das empresas multimédia.

Entratanto, surgiu uma rede interactiva, cujo crescimento alastrava muito mais rapidamente do que a equipa do *7 conseguia vender a ideia. De forma, a recuperar do seu fracasso a Sun incumbiu Gosling de adaptar a linguagem Oak a referida rede interactiva conhecida como Internet. Em 1995, nasceu a linguagem Java, esta surgiu da evolução de Oak. Esta evolução permitia ao Java estar presente tanto na Internet como em diversos dispositivos heterogéneos. Uma vez difundida na Internet, o crescimento da linguagem Java foi totalmente estrondoso.

Desde seu lançamento, em Maio de 1995, a plataforma Java foi adoptada mais rapidamente do que qualquer outra linguagem de programação na história da computação. Em 2004 Java atingiu a marca de 3 milhões de desenvolvedores em todo mundo a linguagem Java continuou crescendo e hoje é uma referência no mercado de desenvolvimento de *software*. Java tornou-se popular pelo seu uso na Internet e hoje possui seu ambiente de execução presente em Web browsers, mainframes, Sistemas Operativos, celulares, palmtops e cartões inteligentes, entre outros.

2.4. JavaServer Pages

JavaServer Pages (JSP) é a tecnologia Java que permite o desenvolvimento dinâmico de documentos HTML, XML entre outros em reposta a um pedido de um cliente Web. Esta tecnologia permite que código em Java seja inserido em código estático.

As funcionalidades e livrarias internas de JSP podem ser activadas ao serem invocadas como se fossem *tags* XML e funcionam como se fossem extensões de HTML ou XML. Estas também podem ser usadas como extensões de capacidades de um servidor Web.

O compilador JSP compila as páginas JSP em *Java Servlets*, podendo estes ser ainda compilados no compilador Java ou ainda serem transformados em *bytecode* para serem mais facilmente interpretados.

2.5. Javascript

A linguagem *JavaScript* foi originalmente criada por Brendan Eich da Netscape com o nome de código *Mocha*, mais tarde veio a chamar-se LiveScript e finalmente *JavaScript*. O nome deu inicialmente muita controvérsia visto que a sua semelhança com a designação Java dava a entender que esta linguagem era uma versão alterada da linguagem com esse nome.

Em Dezembro de 1995 surgiu a *2.0B3* introduzida no navegador Web *Netscape*, em 1996 a Microsoft lança uma versão de *Javascript* com o nome *JScript.*

Nesse mesmo ano, 1996, a Nestcape enviou a linguagem *Javascript* para a Ecma International[3] para que fosse então tornada num *standard*..

2.6. SQL

A *Structured Query Language*, (SQL), (Wikipedia 2008) é uma linguagem de pesquisa declarativa para bases de dados. Muitas das características originais do SQL foram inspiradas na álgebra relacional.

O SQL foi desenvolvido originalmente no início dos anos 70 nos laboratórios da IBM em *San Jose*, dentro do projecto *System R*. Considerada uma linguagem padrão pela simplicidade, por ser declarativa e pela forma clara como apresenta os dados.

Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialectos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem. Esta tarefa foi realizada pela *American National Standards Institute* (ANSI) em 1986 e ISO em 1987.

A linguagem SQL baseia-se em expressões:

- **DML - Linguagem de Manipulação de Dados:** - Primeiro há os elementos da DML (Data Manipulation Language). A DML é um subconjunto da linguagem usada para selecionar, inserir, atualizar e apagar dados.
- **DDL - Linguagem de Definição de Dados: -** O segundo grupo é a DDL (Data Definition Language). Uma DDL permite ao usuário definir tabelas novas e elementos associados. A maioria dos bancos de dados de SQL comerciais tem extensões proprietárias no DDL.
- **DCL - Linguagem de Controle de Dados: -** O terceiro grupo é o DCL (Data Control Language). DCL controla os aspectos de autorização de dados e licenças de usuários para controlar quem tem acesso para ver ou manipular dados dentro do banco de dados.
- **DTL - Linguagem de Transacção de Dados: -** O grupo DTL (Data Transaction Language) permite o controlo de transacções.

Apesar de ser uma linguagem padrão, os diferentes dialectos omitem características básicas de apoio para Standard SQL, tais como a data ou a hora do tipos de dados, preferindo variações próprias. Como resultado, SQL raramente podem ser transferidos entre sistemas de banco de dados sem modificações.

Há várias razões para essa falta de portabilidade entre sistemas de banco de dados:

A complexidade e dimensão do padrão SQL significa que a maior parte das bases de dados não se aplicar a todo o tipo. Como normas não especificas, comportamento do banco de dados em diversos domínios importantes (por exemplo, índices, arquivos de armazenamento...), deixando aos que implementam o banco de dados para decidir o seu comportamento.

O padrão SQL especifica precisamente que a sintaxe conforme um sistema de base de dados deve executar. No entanto, o padrão da especificação da semântica da linguagem construída, é de menor definição, levando a áreas de ambiguidade.

2.7. Oracle e PL/SQL

O *Oracle* é um SGBD (Sistema de Gestão de Bases de Dados) que surgiu no fim dos anos 70, quando Larry Ellison vislumbrou uma oportunidade da qual outras companhias não se tinham percebido, quando encontrou uma descrição de um protótipo funcional de um banco de dados relacional e descobriu que nenhuma empresa tinha se empenhado em comercializar essa tecnologia.

Ellison e os co-fundadores da Oracle Corporation, Bob Miner e Ed Oates, perceberam que havia um tremendo potencial de negócios no modelo de banco de dados relacional tornando assim a maior empresa de software empresarial do mundo.

O Oracle 9i foi pioneiro no suporte ao modelo web. O Oracle 10g, mais recente, se baseia na tecnologia de *grid.* Recentemente fora lançado o Oracle 11g que veio com melhorias em relação ao Oracle 10g.

Além da base de dados, a Oracle desenvolve uma suíte de desenvolvimento chamada de Oracle Developer Suite, utilizada na construção de programas de computador que interagem com a sua base de dados.

A Oracle também criou a linguagem de programação PL/SQL, utilizada no processamento de transações.

PL/SQL (acrónimo para a expressão inglesa *Procedural Language/Structured Query Language*) é uma extensão da linguagem padrão SQL para o SGBD Oracle da Oracle Corporation. É uma Linguagem Procedural da Oracle, estendida da SQL.

Permite que a manipulação de dados seja incluída em unidades de programas. Blocos de PL/SQL são passados e processados por uma PL/SQL Engine que pode estar dentro de uma ferramenta Oracle ou do Server. A PL/SQL Engine filtra os comandos SQL e manda individualmente o comando SQL para o SQL Statement Executor no Oracle Server, que processa o PL/SQL com os dados retornados do Server.

É a linguagem básica para criar programas complexos e poderosos, não só no banco de dados, mas também em diversas ferramentas Oracle.

Antes de 1991 a única forma de usar construções procedurais com o SQL era usar PRO*C. Foi onde as instruções SQL do Oracle foram embutidas em código C. O código C era pré-compilado para converter as instruções SQL em chamadas de bibliotecas.

2.8. Microsoft SQL Server e T-SQL

A base do código do Microsoft SQL Server (anterior à versão7.0) foi originalmente incluída no Sybase SQL Server e foi assim que a Microsoft entrou no mercado das bases de dados entrando em competição directa com a Oracle, a IBM e mais tarde a própria Sybase.

A versão 1.0 foi criada em parceria com a Sybase em 1989 para OS/2 e era muito semelhante ao Sybase SQL Server 3.0 para o UNIX. Mais tarde, a versão 6.0 foi a primeira versão após a separação da Sybase e da Microsoft uma vez que tinham objectivos de mercado diferentes. Depois disto, a Microsoft adquiriu todas as versões do SQL Server e a Sybase mudou o nome do produto para *Adaptive Server Enterprise*.

Quando estas empresas se reuniram para a criação do Microsoft SQL Server decidiram criar também uma linguagem à qual deram o nome *Transact-SQL* (T-SQL).

3. Descrição do Sistema

Em plena era informação, o mais valioso é isso mesmo, a informação. Tendo isto em conta, as estruturas e os sistemas que estão associados ao tratamento, armazenamento e criação de informação sofrem um planeamento mais pormenorizado.

Através da análise de sistema, é permitido aos desenvolvedores estudar a manipulação de informação das suas por parte das suas criações e de como a interacção das mesmas com os utilizadores ocorrem.

Neste caso, através de UML, será realizada a análise funcional da aplicação desenvolvida, especificando e documentando o sistema de informação a ser criado. Neste caso foram utilizados os seguintes diagramas: diagramas de casos de uso, diagramas de sequência, diagramas de actividades e diagramas de classes.

Esta parte da criação da aplicação foi criada em conjunto com o aluno Manuel Vieira.

3.1. Casos de uso

Os diagramas de Casos de Uso identificam os serviços, utilizadores e fronteiras de um sistema. Assim, facilmente poderão identificar-se os serviços associados aos diferentes utilizadores bem como os seus processos inerentes.

Devido à elevada quantidade de casos de usos, os mesmos serão ilustrados em dois diagramas. O primeiro diagrama representará os casos de uso responsáveis pela gestão e no segundo serão agrupados os casos de uso com vista às consultas. Ambos dão-nos uma visão geral do sistema das funcionalidades da aplicação e das situações em que é feito o controlo de acesso aos utilizadores, onde os utilizadores são separados em dois tipos: Administrador e Funcionário, tendo o utilizador do tipo funcionários o acesso limitado a algumas funcionalidades da aplicação.

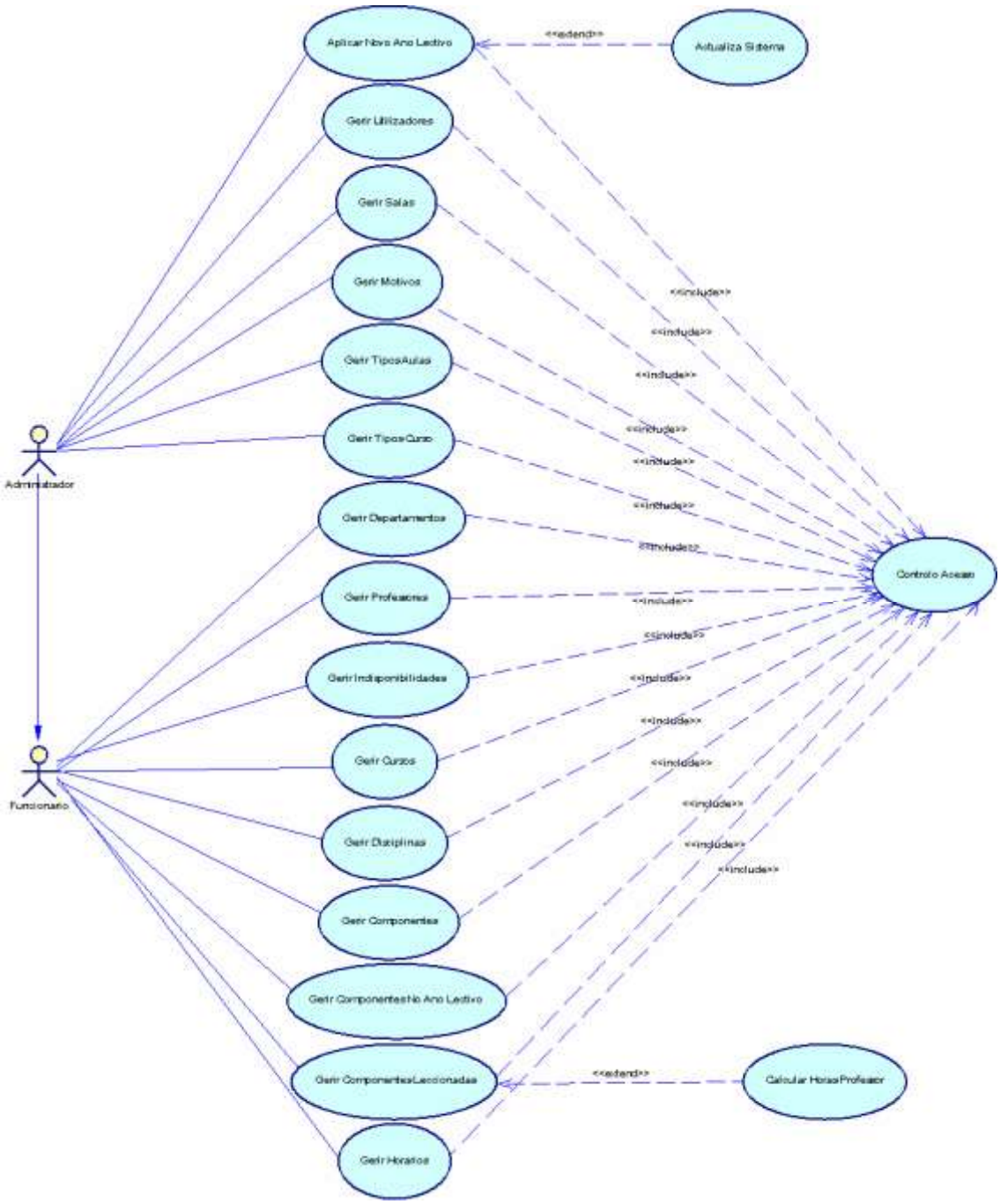

Figura 3.1– Diagrama Casos de Uso "Gestão"

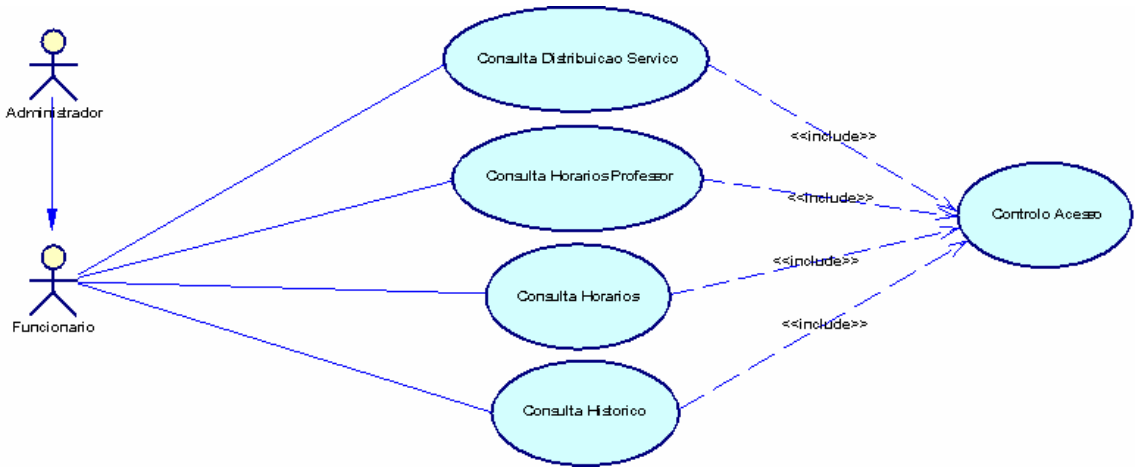

Figura 3.2– Diagrama Casos de Uso "Consultas"

Na figura 3.2 encontram-se ilustrados os casos de uso responsáveis pelas consultas e impressão de resultado

3.1.1. Descrição estruturada dos Casos de Uso

Serão agora descritos de forma estruturada todos os casos de uso visíveis nos diagramas de casos de uso (figura 3.1 e 3.2). Em todos os casos de uso é descrita uma pré - condição que indica o que deve existir inicialmente para que o cenário descrito seja seguido com sucesso.

Na descrição de um caso de uso pressupõe-se que estão reunidas todas as condições que garantem que tudo corre bem, sendo um cenário onde não surgem problemas, denominado como o cenário principal. [3]

Na tabela 3.1 está descrito o caso de uso "Controlo de Acesso", responsável por atribuir o acesso aos utilizadores registados.

Tabela 3.1- Descrição Estruturada do Caso de Uso "Controlo Acesso"

Controlo Acesso	
Objectivo	Entrar no Sistema
Descrição	1 - O Caso de Uso começa quando o utilizador insere o seu username e pasword. 2 - É verificado se o utilizador existe na BD e qual o seu tipo. 3- É verificado o estado do utilizador.

Tabela 3.2- Descrição Estruturada do Caso de Uso "Gerir Salas"

Gerir Salas	
Objectivos	Inserir, modificar e eliminar Salas.
Pré - Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Salas" dentro do ecrã "Outros". 2 - Aceder à área "Salas" para inserir, modificar ou eliminar uma sala. 3 - Serão listadas as salas existentes. 4 - Caso o utilizador tente inserir uma nova sala ou modificar uma sala existente com um nome de sala já existente o sistema avisa a impossibilidade de efectuar essa operação. 5 - A sala tem um estado associado a ela, sendo ele disponível ou indisponível e obrigatório. Pode ser alterado a qualquer altura. 6- As operações são confirmadas nos botões "Nova", "Modificar" e "Eliminar". 7 - Caso o utilizador tente eliminar uma sala que esteja atribuída a uma componente no ano lectivo o sistema avisa a impossibilidade de efectuar essa operação.
Casos de uso relacionados	Include: Controlo de Acesso.

Tabela 3.3- Descrição Estruturada do Caso de Uso "Gerir Tipos Curso"

Gerir Tipos Curso	
Objectivos	Inserir, modificar e eliminar tipos de curso.
Pré – Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Tipos de Curso" dentro do ecrã "Menu Principal". 2. Aceder à área "Tipos de Curso" para inserir, modificar ou eliminar um tipo de curso. 3. São listados numa os tipos de curso existentes. 4. Caso o utilizador tente inserir um novo tipo de curso ou modificar um tipo de curso existente com um nome de tipo de curso já existente o sistema avisa a impossibilidade de efectuar essa operação. 5. As operações são confirmadas nos botões "Novo", "Alterar" e "Eliminar". 6. Caso o utilizador tente eliminar um tipo de curso que esteja atribuído a um curso o sistema avisa a impossibilidade de efectuar essa operação.
Casos de uso relacionados	Include: Controlo de Acesso.

Tabela 3.4- Descrição Estruturada do Caso de Uso "Gerir Tipo Aulas"

Gerir Tipo Aulas	
Objectivos	Inserir, modificar e eliminar tipos de aulas.
Pré - Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador.
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Tipos de Aulas" dentro do Ecrã "Menu Principal". 2 - Aceder à área "Tipo Aulas" para inserir, modificar ou eliminar um tipo de aulas. 3 - Serão listados os tipos de aulas existentes 4 - Caso o utilizador tente inserir um novo tipo de aula ou modificar um tipo de aula existente com um nome já existente o sistema avisa a impossibilidade de efectuar essa operação 5 - As operações são confirmadas nos botões "Novo", "Modificar" e "Eliminar". 6 - Caso o utilizador tente eliminar um tipo de aula que esteja atribuído a uma componente o sistema avisa a impossibilidade de efectuar essa operação.
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.5- Descrição Estruturada do Caso de Uso "Gerir Motivos"

Gerir Motivos	
Objectivos	Inserir, modificar e eliminar motivos.
Pré - Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador.
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Motivos" dentro do ecrã" Outros". 2 - Aceder à área "Motivos" para inserir, modificar ou eliminar um motivo. 3 - Serão listados os motivos existentes, organizados pela tabela a que pertencem 4 - Todos os motivos têm de estar associados a uma tabela. 5 - As tabelas existentes serão exibidas numa ComboBox. 6 - Caso o utilizador tente inserir um novo motivo ou modificar um motivo existente com um nome já existente e associado a uma mesma tabela o sistema avisa a impossibilidade de efectuar essa operação. 7 - As operações são confirmadas nos botões "Novo", "Modificar" e "Eliminar". 8 - Caso o utilizador tente eliminar um motivo que esteja associado a algum registo o sistema avisa a impossibilidade de efectuar essa operação.
Casos de uso relacionados	Include: Controlo de Acesso.

Tabela 3.6- Descrição Estruturada do Caso de Uso "Gerir Utilizadores"

Gerir Utilizadores	
Objectivos	Inserir, modificar e eliminar utilizadores.
Pré - Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Utilizadores" no "Menu Proncipal". 2 - Aceder à área Utilizadores para Inserir, Modificar ou Eliminar um Utilizador. 3 - Serão listados os utilizadores existentes. 4 - Os utilizadores poderão de dois tipos: Administrador e Funcionário. 5 - O tipo de utilizador será exibido numa Combobox. 6 - Caso o Administrador tente inserir um novo utilizador ou modificar um utilizador existente com um username já existente o sistema avisa a impossibilidade de efectuar essa operação. 7 - É o Administrador que atribui a password ao utilizador, 8- As operações são confirmadas nos botões "Novo", "Modificar" e "Eliminar".
Casos de uso relacionados	Include: Controlo de Acesso.

Tabela 3.7 - Descrição Estruturada do Caso de Uso "Gerir Departamentos"

Gerir Departamentos	
Objectivos	Inserir, modificar e eliminar departamentos.
Pré - Condição	O Utilizador terá que ser válido no sistema.
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Departamentos e Professores". 2. Aceder à área "Departamentos" para inserir, modificar ou eliminar um Departamento. 3. Serão listados numa os departamentos existentes. 4. Caso o utilizador tente inserir um novo departamento ou modificar um departamento existente com um nome já existente o sistema avisa da impossibilidade de efectuar essa operação. 5. 5 - Quando um departamento é inserido ou modificado é guardado o ID do utilizador que efectuou a ultima alteração. 6. Quando um departamento é inserido, é guardada a data do sistema como data de inserção. 7. Quando um departamento é modificado, é guardada a data do sistema como data da última alteração. 8. Caso o utilizador tente eliminar um departamento que tenha associado a ele professores, o sistema avisa a impossibilidade de efectuar essa operação. 9. As operações são confirmadas nos botões "Novo", "Modificar" e "Eliminar".
Casos de uso relacionados	Include: Controlo de Acesso.

Tabela 3.8 - Descrição Estruturada do Caso de Uso "Gerir Professores"

Gerir Professores	
Objectivos	Inserir, modificar e eliminar professores.
Pré - Condição	O Utilizador terá que ser válido no sistema.
Descrição:	1. O Caso de Uso começa quando o utilizador escolhe a opção "Professores" no "Menu Principal" para inserir, modificar ou eliminar um Professor 2. Serão listados os professores existentes 3. Os professores poderão ser listados pelo nome e pelo departamento ao qual pertencem 4. Cada professor terá de estar associado a um departamento 5. Os departamentos serão exibidos numa Combo - box 6. Quando um professor e inserido ou modificado é guardado o ID do utilizador que efectuou a ultima alteração 7. Quando um professor é inserido, é guardada a data do sistema como data de inserção 8. Quando um professor é modificado, é guardada a data do sistema como data da última alteração 9. As operações são confirmadas nos botões Inserir, Modificar e Elimina
Casos de uso relacionados.	Include: Controlo de Acesso

Tabela 3.9 - Descrição Estruturada do Caso de Uso "Gerir Indisponibilidades"

Gerir Indisponibilidades	
Objectivos	Inserir, modificar e eliminar indisponibilidades
Pré - Condição	O Utilizador terá que ser válido no sistema
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Departamentos e Professores" 2 - Dentro da área "Professores" aceder à área "indisponibilidades" para inserir, modificar ou eliminar uma indisponibilidade . 3 - Será listadas todas indisponibilidades 4 - Todas a alterações serão removidas sempre que um novo ano lectivo seja criado
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.10 - Descrição Estruturada do Caso de Uso "Gerir Cursos"

Gerir Cursos	
Objectivos	Inserir, modificar e eliminar cursos
Pré - Condição	O Utilizador terá que ser válido no sistema
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Cursos e Disciplinas" 2 - Aceder à área "Cursos" para inserir, modificar ou liminar um Curso 3 - Serão listados numa os cursos existentes 4 - Os cursos poderão ser listados por nome e por tipo de curso 5 - Todos os cursos terão de ter um tipo de curso associado 6 - Os tipos de curso serão exibidos numa combobox 7 - Caso o utilizador tente eliminar um Curso ao qual estejam associadas disciplinas em uso o sistema avisa a impossibilidade de efectuar essa operação 8 - Quando um curso é inserido ou modificado é guardado o ID do utilizador que efectuou a ultima alteração 9- Quando um curso é inserido, é guardada a data do sistema como data de inserção 10 - Quando um curso é modificado, é guardada a data do sistema como data da última alteração
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.11- Descrição Estruturada do Caso de Uso "Gerir Disciplinas"

Gerir Disciplinas	
Objectivos	Inserir, modificar e eliminar disciplinas
Pré - Condição	O Utilizador terá que ser válido no sistema
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Disciplinas" no "Menu Principal" para inserir, modificar ou eliminar uma Disciplina 2. Serão listadas as disciplinas existentes 3. As disciplinas poderão ser listadas por nome, curso ou semestre 4. As disciplinas têm que estar associadas a um curso 5. Os cursos serão exibidos numa Combo - box 6. Caso o utilizador tente eliminar uma disciplina à qual esteja associada uma componente no ano lectivo corrente o sistema avisa a impossibilidade de efectuar essa operação 7. Quando uma disciplina e inserido ou modificado é guardado o ID do utilizador que efectuou a ultima alteração 8. Quando uma disciplina é inserida, é guardada a data do sistema como data de inserção 9. Quando uma disciplina é modificada, é guardada a data do sistema como data da última alteração
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.12- Descrição Estruturada do Caso de Uso "Gerir Componentes"

Gerir Componentes	
Objectivos	Inserir, modificar e eliminar componentes
Pré - Condição	O Utilizador terá que ser válido no sistema
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Componentes" no "Menu Principal" para inserir, modificar ou eliminar uma Componente 2. Serão listadas as componentes respectivas a cada disciplina 3. Caso o utilizador tente eliminar uma componente à qual esteja associada uma componente no ano lectivo corrente o sistema avisa a impossibilidade de efectuar essa operação 4. Quando uma componente é inserida ou modificada é guardado o ID do utilizador que efectuou a última alteração 5. Quando uma componente é inserida, é guardada a data do sistema como data de inserção 6. Quando uma componente é modificada, é guardada a data do sistema como data da última alteração
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.13- Descrição Estruturada do Caso de Uso "Gerir Componentes Ano Lectivo"

Gerir Componentes Ano Lectivo	
Objectivos	Inserir, modificar e eliminar componentes ano lectivo
Pré - Condição	O Utilizador terá que ser válido no sistema
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Componentes no Ano Lectivo" 2. Aceder à área "Componentes Ano Lectivo" para inserir, modificar ou eliminar uma Componente_AL 3. Serão listadas numa datagridview todas as componentes existentes no ano lectivo corrente 4. Cada componente_al terá que possuir uma sala principal e uma prioridade para essa sala bem como uma sala alternativa 5. Caso o utilizador pretenda inserir uma nova componente_al já existente o sistema avisa a impossibilidade de efectuar essa operação 6. Caso o utilizador pretenda eliminar uma componente no ano lectivo que esteja associada a um horários em uso no ano lectivo corrente o sistema avisa a impossibilidade de efectuar essa operação 7. Quando uma componente_al é inserida ou modificada é guardado o ID do utilizador que efectuou a ultima alteração 8. Quando uma componente_al é inserida, é guardada a data do sistema como data de inserção 9. Quando uma componente_al é modificada, é guardada a data do sistema como data da última alteração 10. As componentes_al são criadas no corrente ano lectivo
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.14- Descrição Estruturada do Caso de Uso "Gerir Componentes Leccionadas"

Gerir Componentes Leccionadas	
Objectivos	Inserir, modificar e eliminar componentes leccionadas
Pré - Condição	O Utilizador terá que ser válido no sistema Para atribuir uma componente a um professor, têm de existir o professor e a componente no ano lectivo Para eliminar ou modificar determinada componente leccionada esta terá de existir
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "atribuir professor à componente" 2. Aceder à área "atribuir professor à component"e para inserir, modificar ou eliminar um professor atribuído a uma componente 3. As componentes leccionadas são listadas 4. As turmas criadas para cada componente leccionada não podem exceder o número de turmas da componente no ano lectivo 5. O número de horas da componente leccionada tem que ser igual ao número de horas da componente 6. Caso o utilizador pretenda eliminar uma componente leccionada que esteja associada a um horário em uso no ano lectivo corrente o sistema avisa a impossibilidade de efectuar essa operação 7. Quando uma componente leccionada é inserida ou modificado é guardado o ID do utilizador que efectuou a ultima alteração 8. Quando uma componente leccionada é inserida, é guardada a data do sistema como data de inserção 9. Quando uma componente leccionada é modificada, é guardada a data do sistema como data da última alteração
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.15- Descrição Estruturada do Caso de Uso "Gerir Horarios"

Gerir Horários	
Objectivos	Inserir, modificar e eliminar horários
Pré - Condição	O Utilizador tem que ser válido no sistema
Descrição	1. O Caso de Uso começa quando o utilizador escolhe a opção "Horarios" 2. Aceder à área" Horários" para inserir, modificar, eliminar ou visualizar um horário 3. Após a selecção do curso, ano, semestre e turma, preencher o horário seguindo as indicações do sistema se o utilizador assim o desejar 4. Os Horários podem ser impressos 5. O Utilizador pode alterar a sala predestinada da componente, especificada na altura de criação da componente no ano lectivo por uma outra.
Casos de uso relacionados	Include: Controlo de Acesso

Tabela 3.16- Descrição Estruturada do Caso de Uso "Aplicar Novo Ano Lectivo"

Aplicar Novo Ano Lectivo	
Objectivo	Inserir um novo ano lectivo
Pré - Condição	O Utilizador terá que ser válido no sistema e do tipo Administrador
Descrição	1 - O Caso de Uso começa quando o utilizador escolhe a opção "Novo Ano Lectivo" 2 - Aceder à área "Novo Ano Lectivo" para inserir um novo ano lectivo 3 - A criação de um novo ano lectivo é automática, não tendo o utilizador possibilidade de definir o ano lectivo a criar
Casos de uso relacionados	Include: Controlo de Acesso

3.2. Diagramas de Sequência

Os diagramas de sequência documentam as interacções que estão por destras das funcionalidades dos casos de usos.

Assim é ilustrado, estruturadamente, o desenvolvimento dos casos de uso através de interacções entre objectos.

Não será exibido o diagrama de sequência para o caso de uso "Controlo Acesso" uma vez que é efectuado em todos os diagramas de sequência.

Serão então demonstrados os diagramas de sequência para os casos de uso dos diagramas apresentados anteriormente neste capítulo.

3.2.1. Diagrama de Sequência - "Aplica Novo Ano Lectivo"

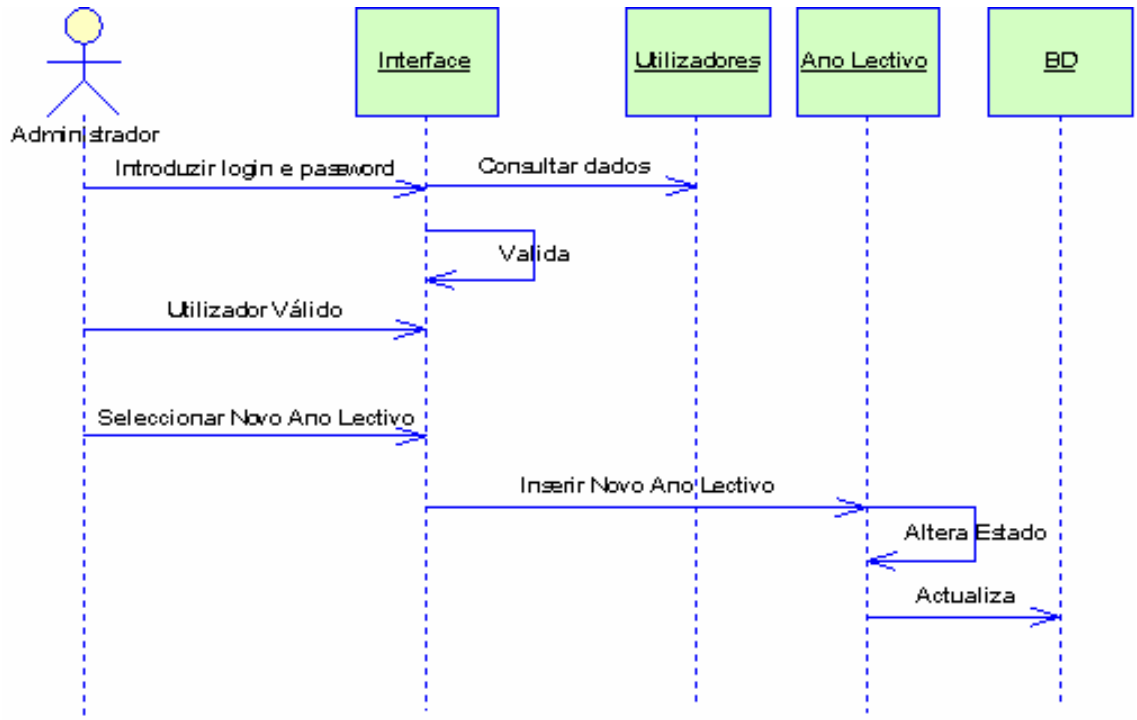

Figura 3.3- Diagrama de Sequência" Aplica Novo Ano Lectivo"

3.2.2. Diagrama de Sequência - "Gerir Utilizadores"

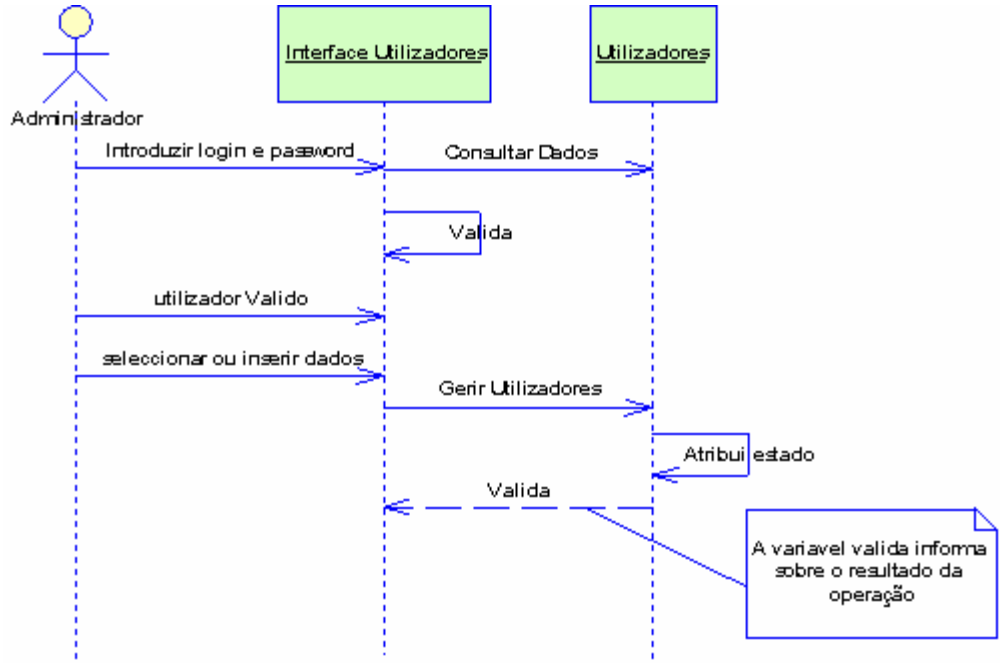

Figura 3.4- Diagrama de Sequência "Gerir utilizadores"

3.2.3. Diagrama de Sequência - "Gerir Salas"

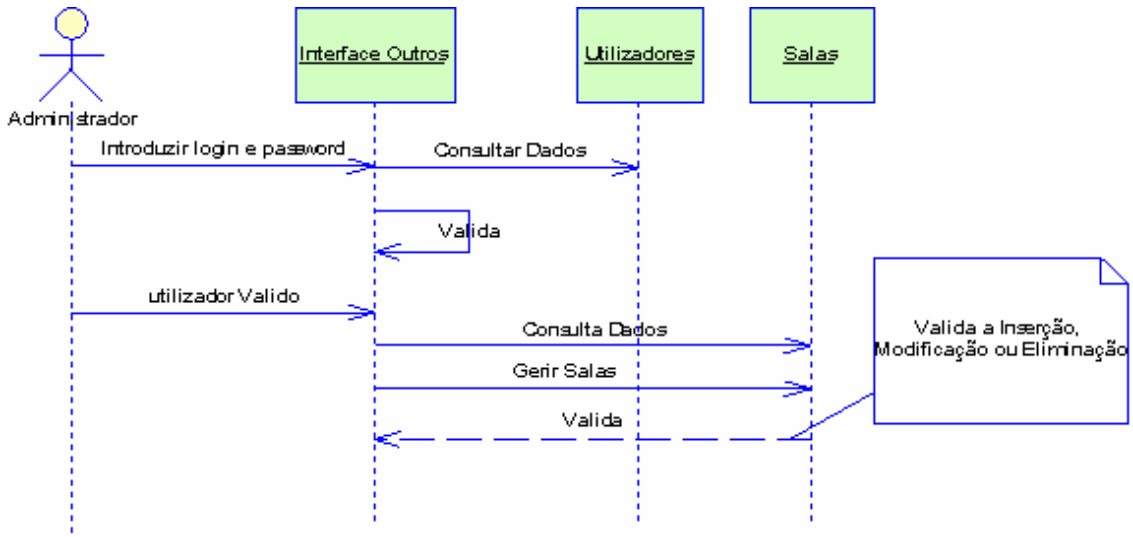

Figura 3.5- Diagrama de Sequência "Gerir Salas"

Os diagramas de sequência para os casos de uso "Gerir Tipos Curso"e "Gerir Tipo Aulas", não serão exibidos, pois são idênticos ao diagrama da figura 3.5.

3.2.4. Diagrama de Sequência - "Gerir Motivos"

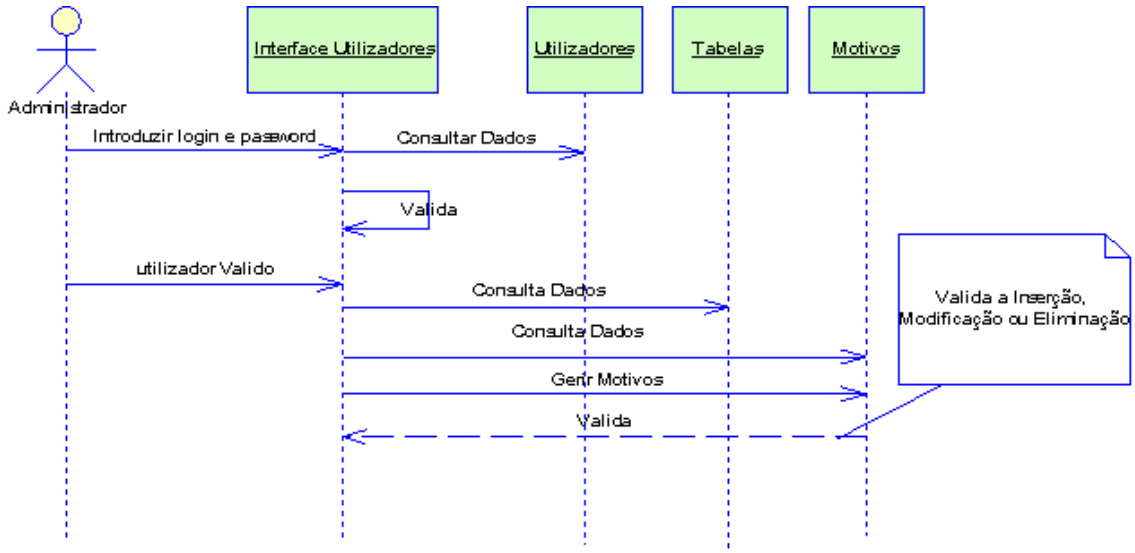

Figura 3.6- Diagrama de Sequência "Gerir Motivos"

3.2.5. Diagrama de Sequência - "Gerir Departamentos"

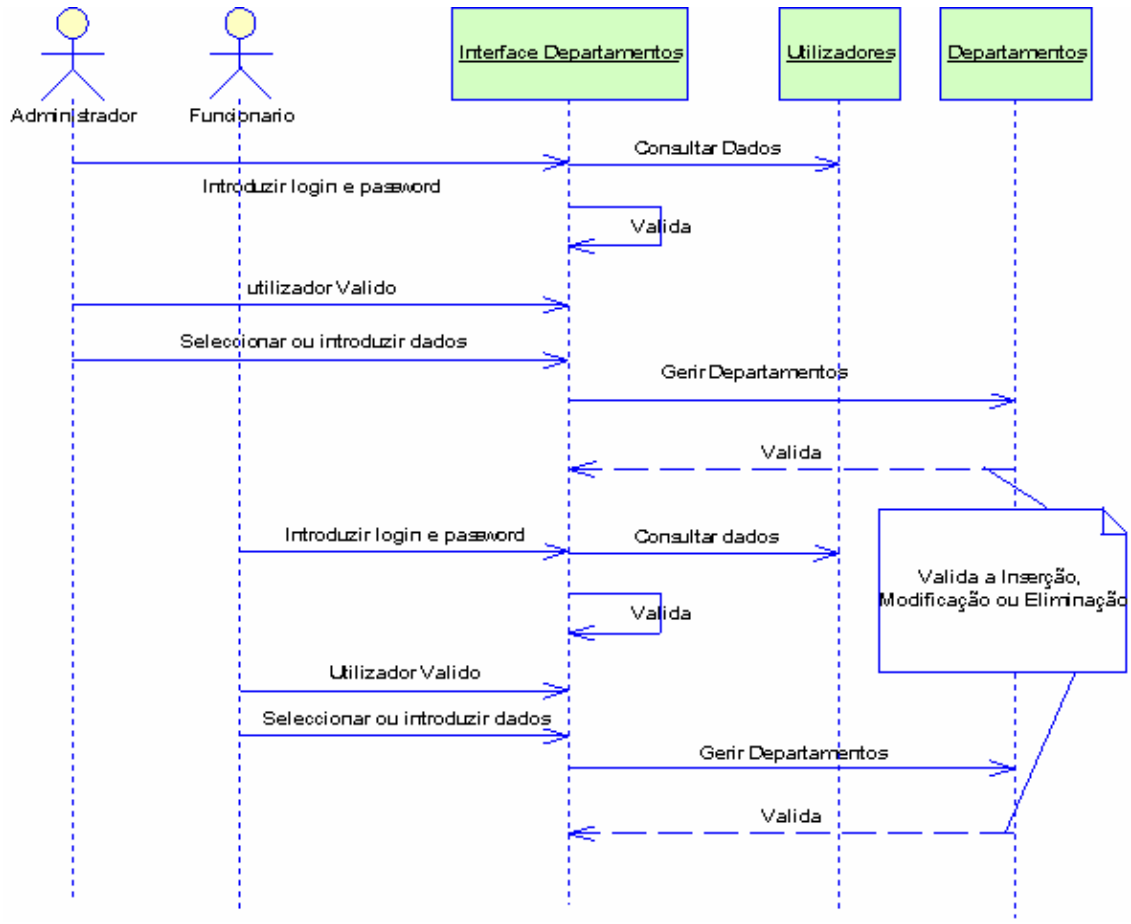

Figura 3.7- Diagrama de Sequência "Gerir Departamentos"

Os diagramas de sequência para o caso de uso "Gerir Professores" e "Gerir Curso" não serão exibidos, pois são semelhantes ao da figura 3.7.

3.2.6. Diagrama de Sequência - "Gerir Disciplinas"

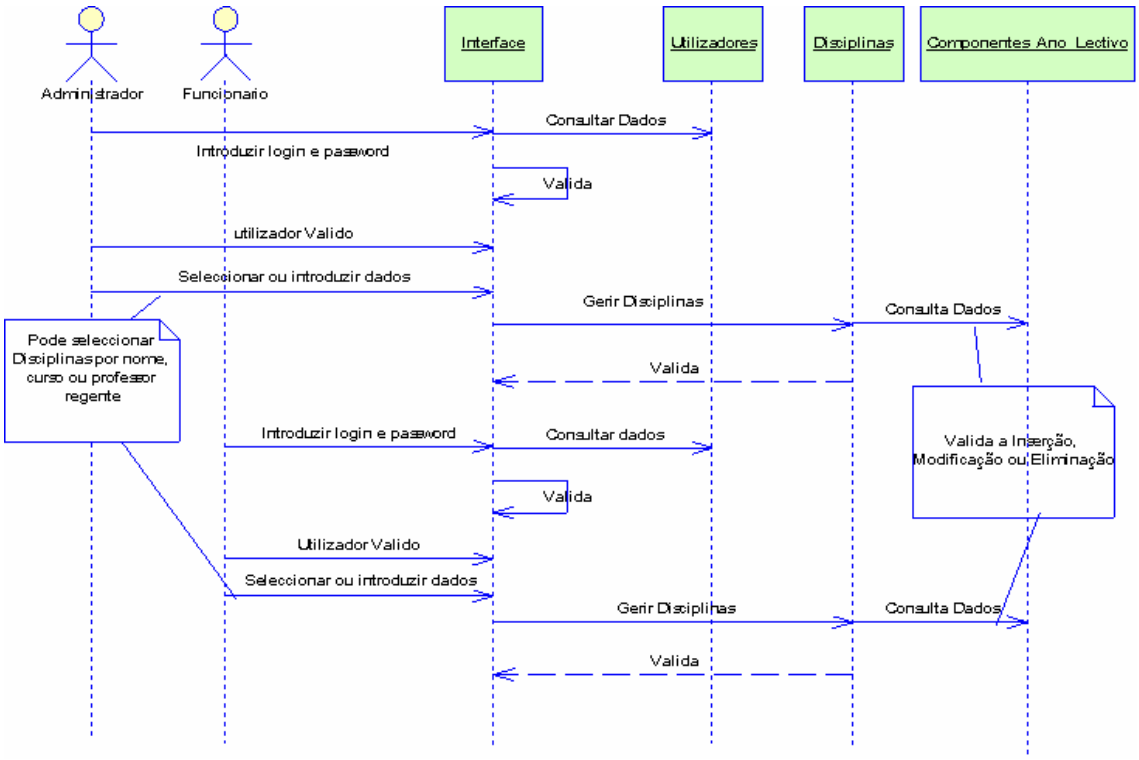

Figura 3.8- Diagrama de Sequência "Gerir Disciplinas"

3.2.7. Diagrama de Sequência - "Gerir Componentes"

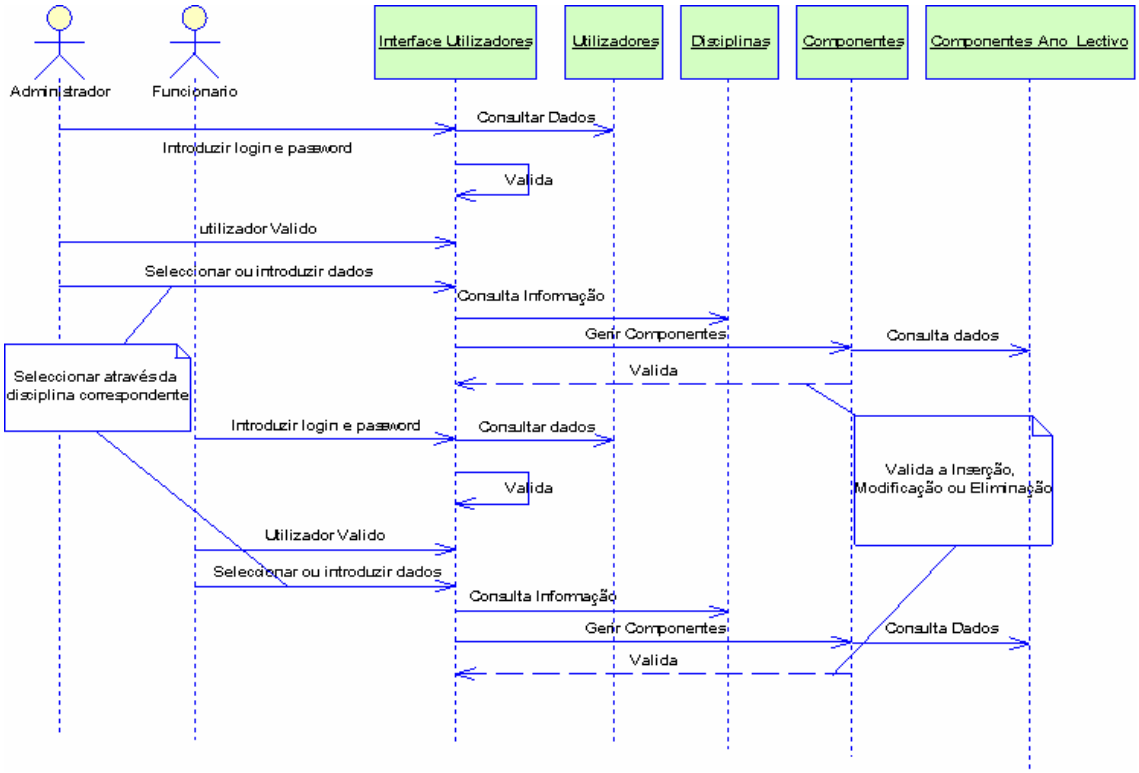

Figura 3.9- Diagrama de Sequência "Gerir Componentes"

3.2.8. Diagrama de Sequência - "Gerir Componentes No Ano Lectivo"

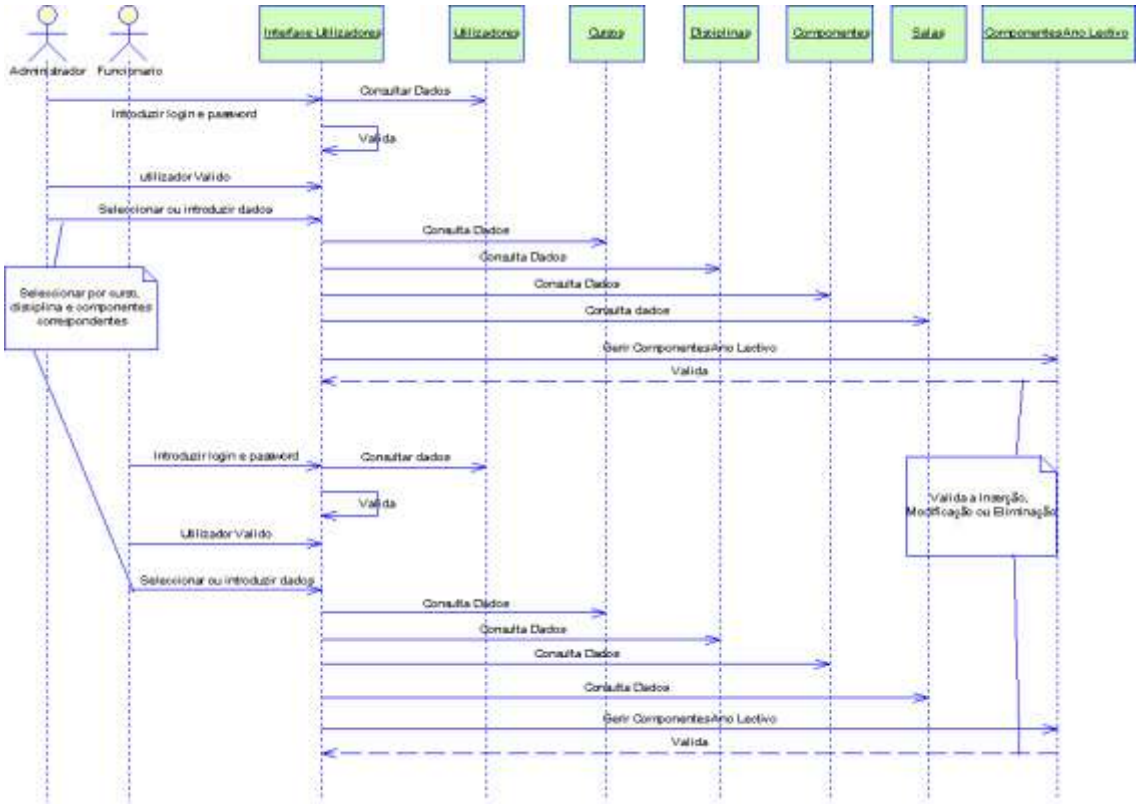

Figura 3.10 - Diagrama de Sequência "Gerir Componentes No Ano Lectivo"

3.2.9. Diagrama de Sequência - "Gerir Componentes leccionadas"

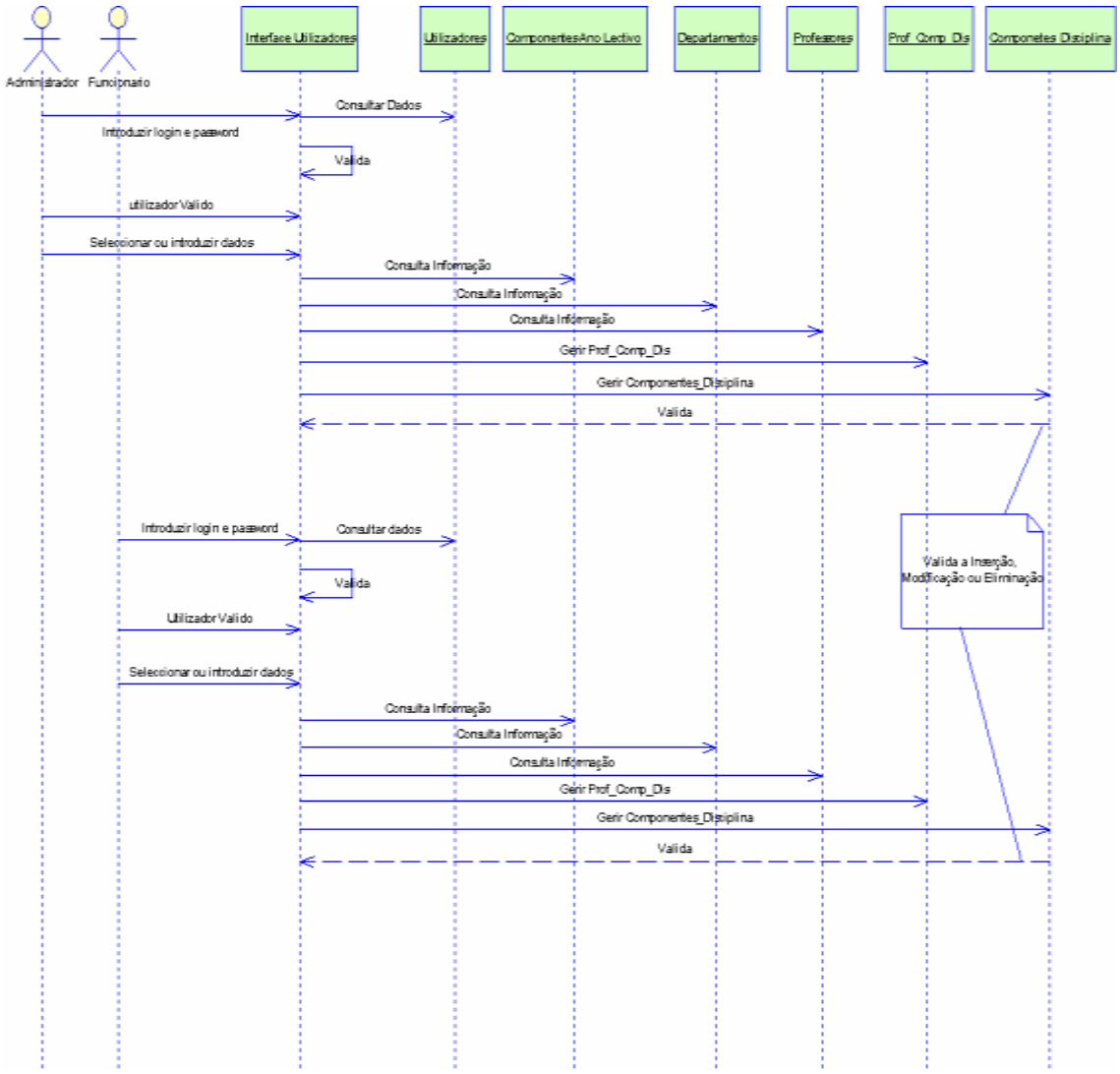

Figura 3.11 - Diagrama de Sequência "Gerir Componentes Leccionadas"

3.2.10. **Diagrama de Sequência - "Gerir Horários**"

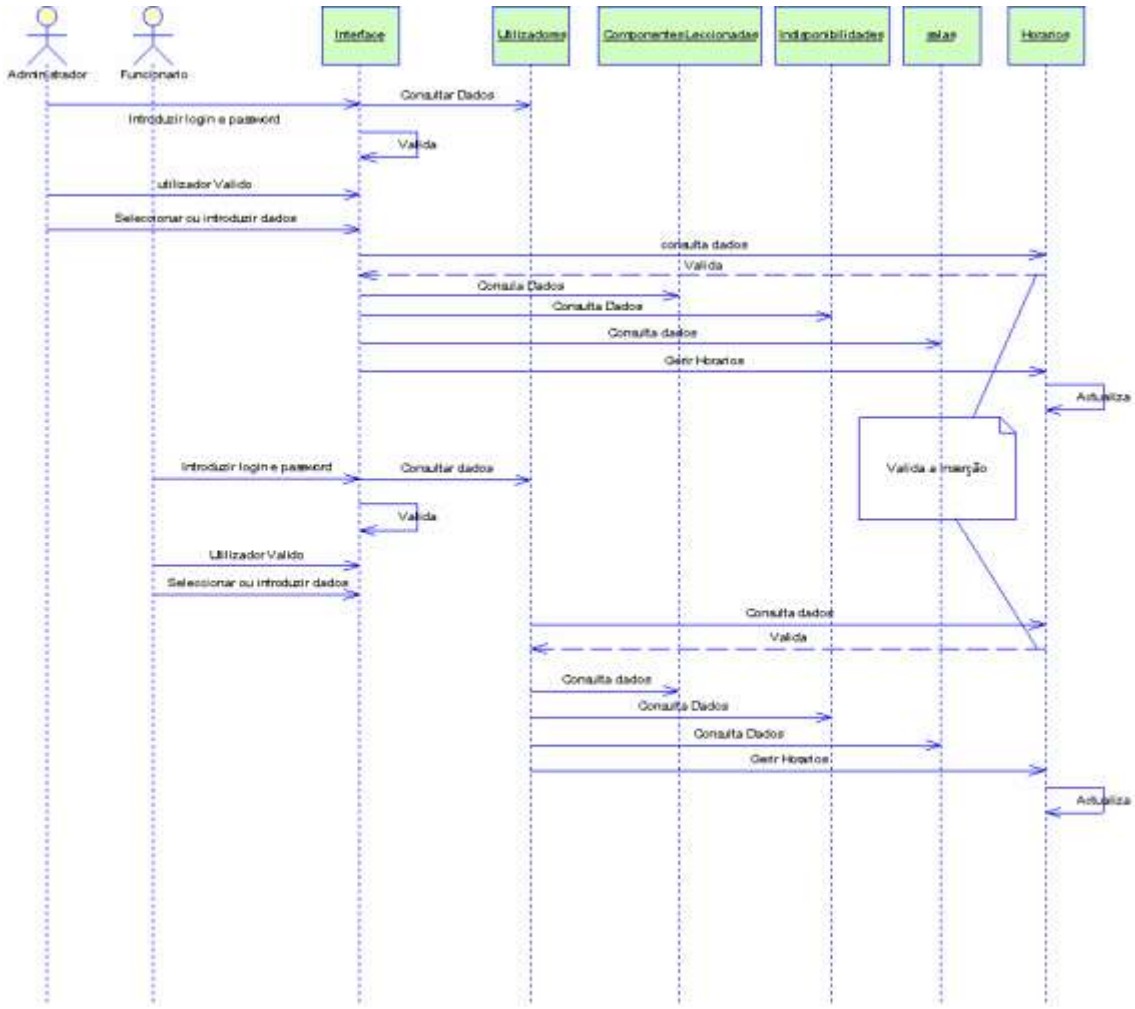

Figura 3.12 - Diagrama de Sequência "Gerir Horários"

3.2.11. Diagrama de Sequência - "Gerir Indisponibilidades"

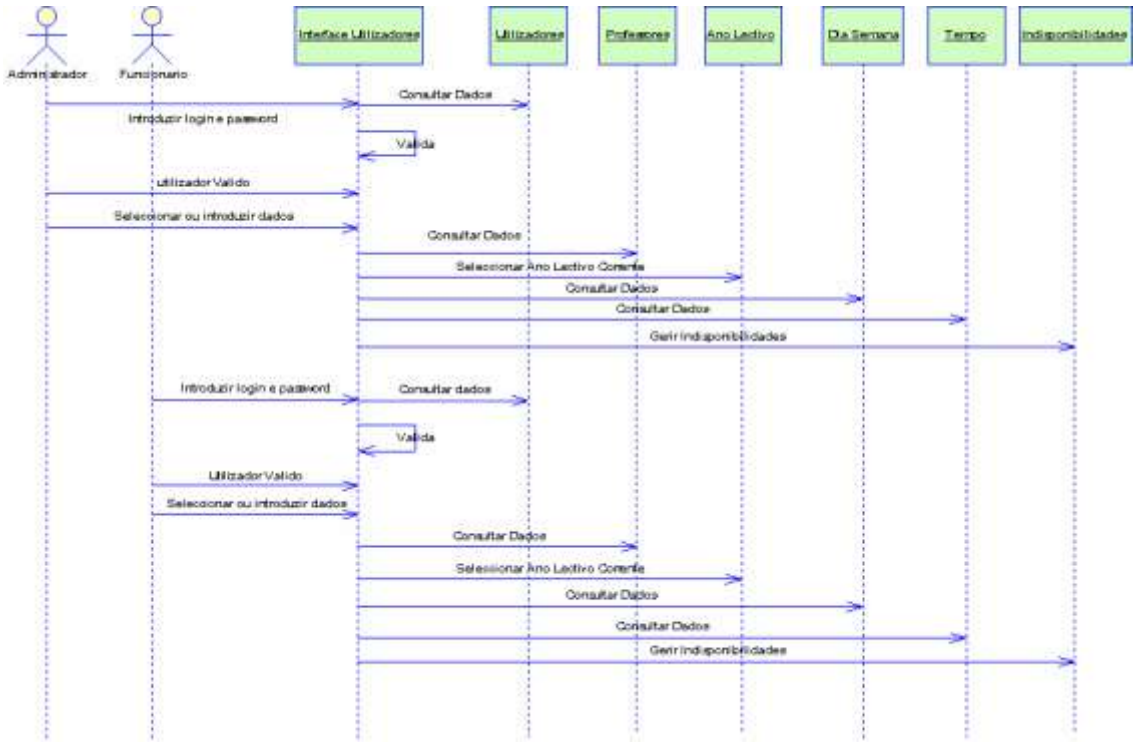

Figura 3.13 - Diagrama de Sequência "Gerir Indisponibilidades"

3.2.12. Diagrama de Sequência - "Consultar Horários"

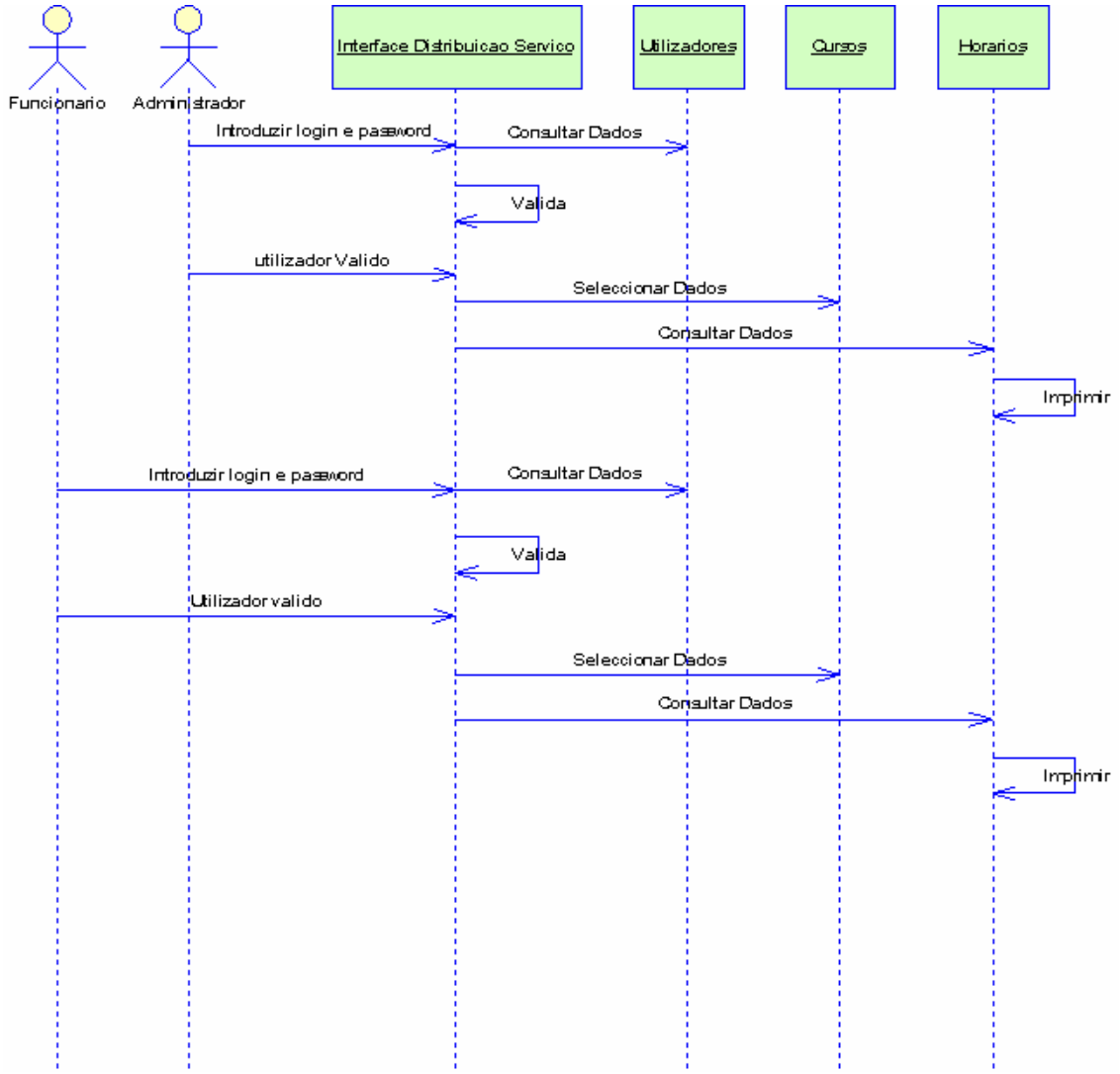

Figura 3.14 - Diagrama de Sequência "Consulta Horários"

3.2.13. Diagrama de Sequência - "Criar Horários"

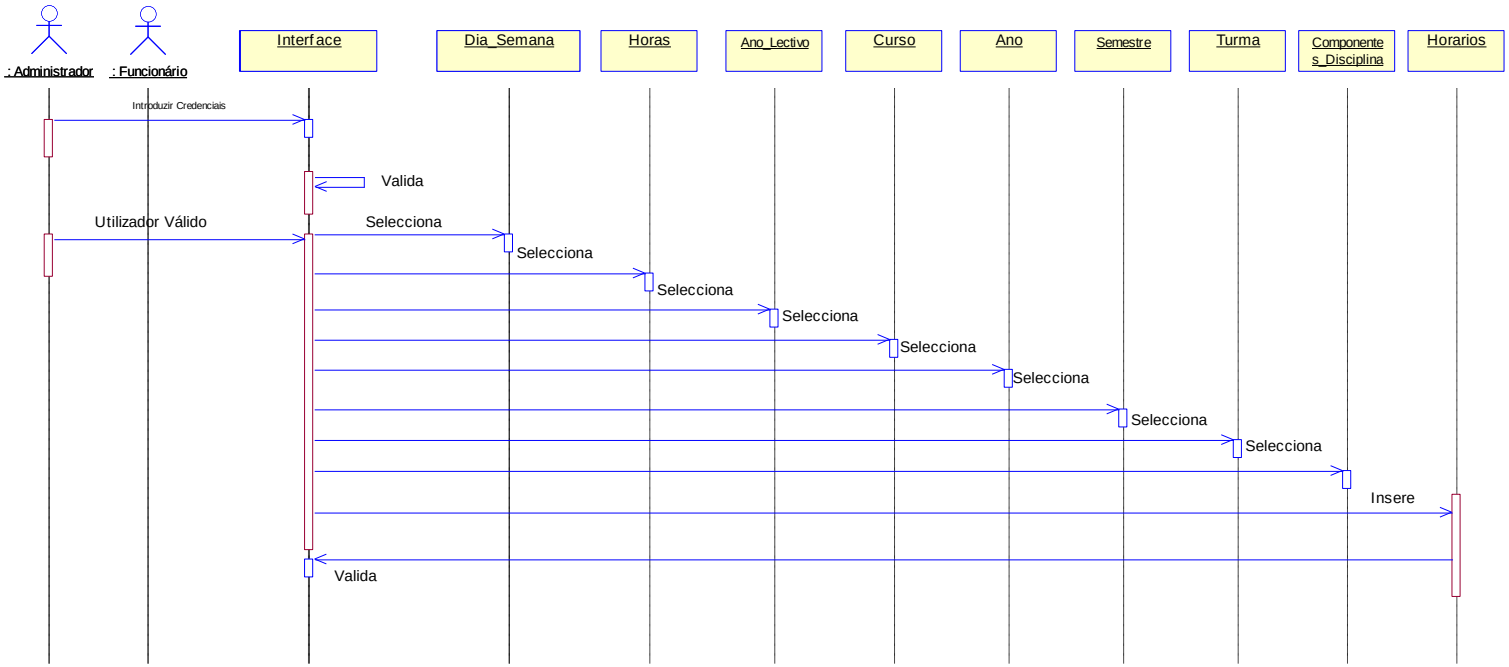

Figura 3.15 - Diagrama de Sequência "Insere Horários"

3.3. Diagramas de Actividade

Estes diagramas de actividade descrevem as actividades de sistema que ocorrem paralelamente envolvendo assim diversos casos de uso.

Como para alguns dos casos de uso, a descrição estruturada e os diagramas de sequência não foram totalmente elucidativos, foram construídos diagramas de actividade numa tentativa de clarificar as acções que ocorrem.

Os Diagramas de Actividades que serão apresentados de seguida são: "Gerir Horários" e "Gerir Componentes Leccionadas".

3.3.1. **Diagrama de Actividades - "Gerir Horários**"

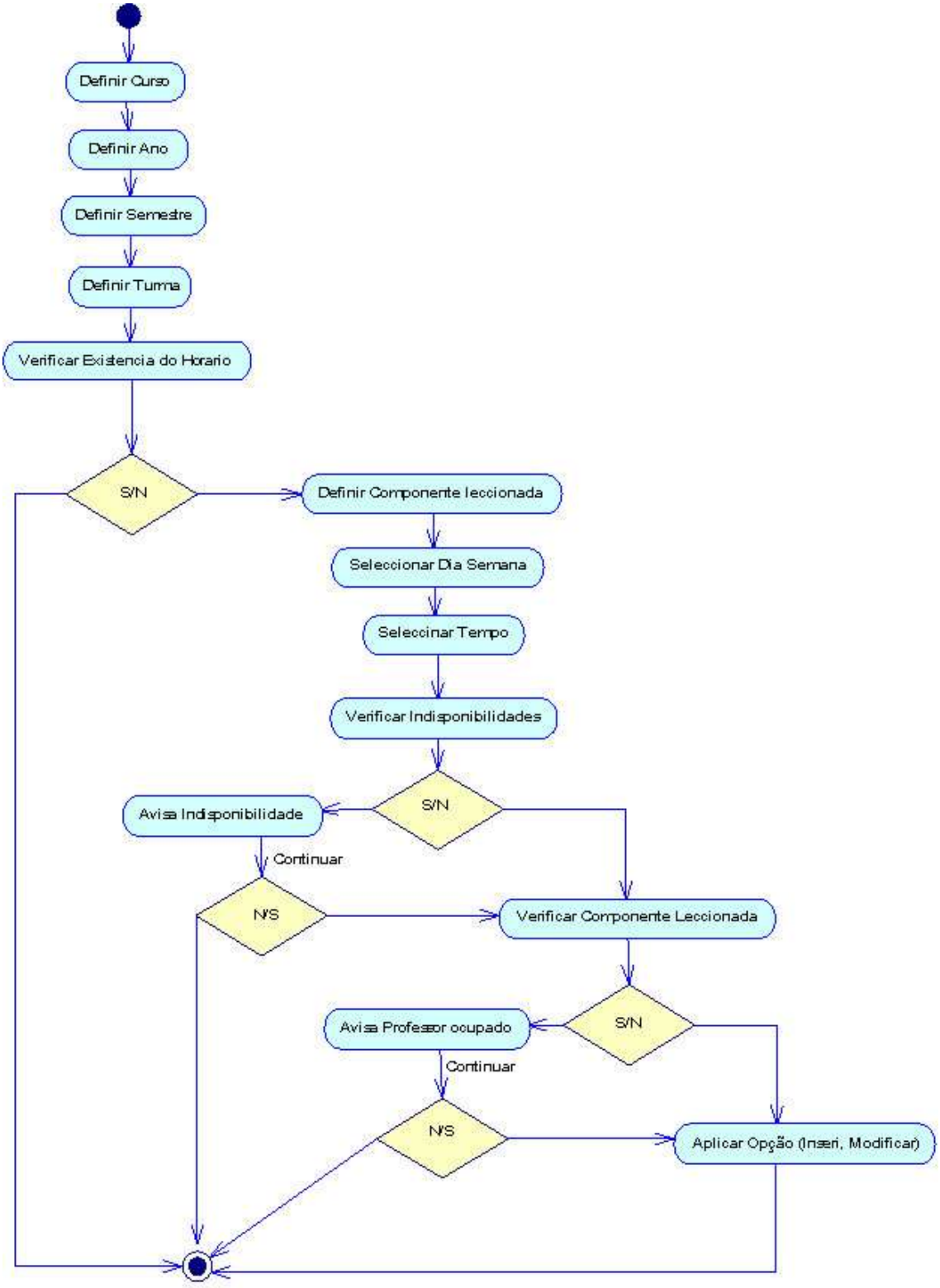

Figura 3.16 - Diagrama de Actividades "Gerir Horários"

3.3.2. Diagrama de Actividades - "Gerir Componentes Leccionadas"

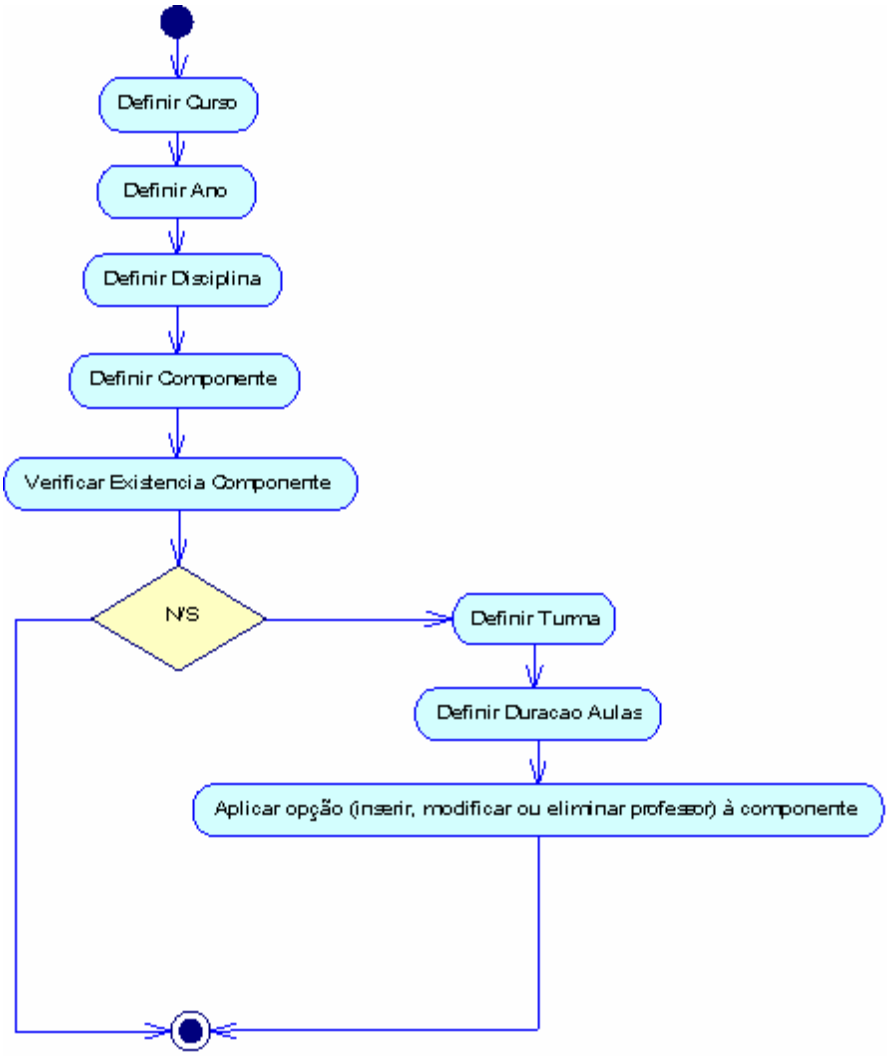

Figura 3.17- Diagrama de Actividades "Gerir Componentes Leccionadas"

3.4. Diagrama de Classes

A criação de um modelo de classes resulta de um processo de abstracção através do qual se identificam os objectos (entidades e conceitos) relevantes no contexto que se pretende modelar e se procuram descrever características comuns em termos de propriedades (atributos) e de comportamentos (operações). A essa descrição genérica dá-se o nome de classe [3].

O Diagrama de Classes é uma das técnicas mais utilizadas no desenvolvimento orientado a objectos, pois define a estrutura do sistema a desenvolver. Em baixo na figura 3.18, esta representado o diagrama de classes do MN Horários. Nas classes, por baixo dos atributos, são definidas as operações utilizadas pelas classes. As operações são a representação lógica do comportamento de um objecto, consistindo em acções efectuadas por ou sobre um objecto.

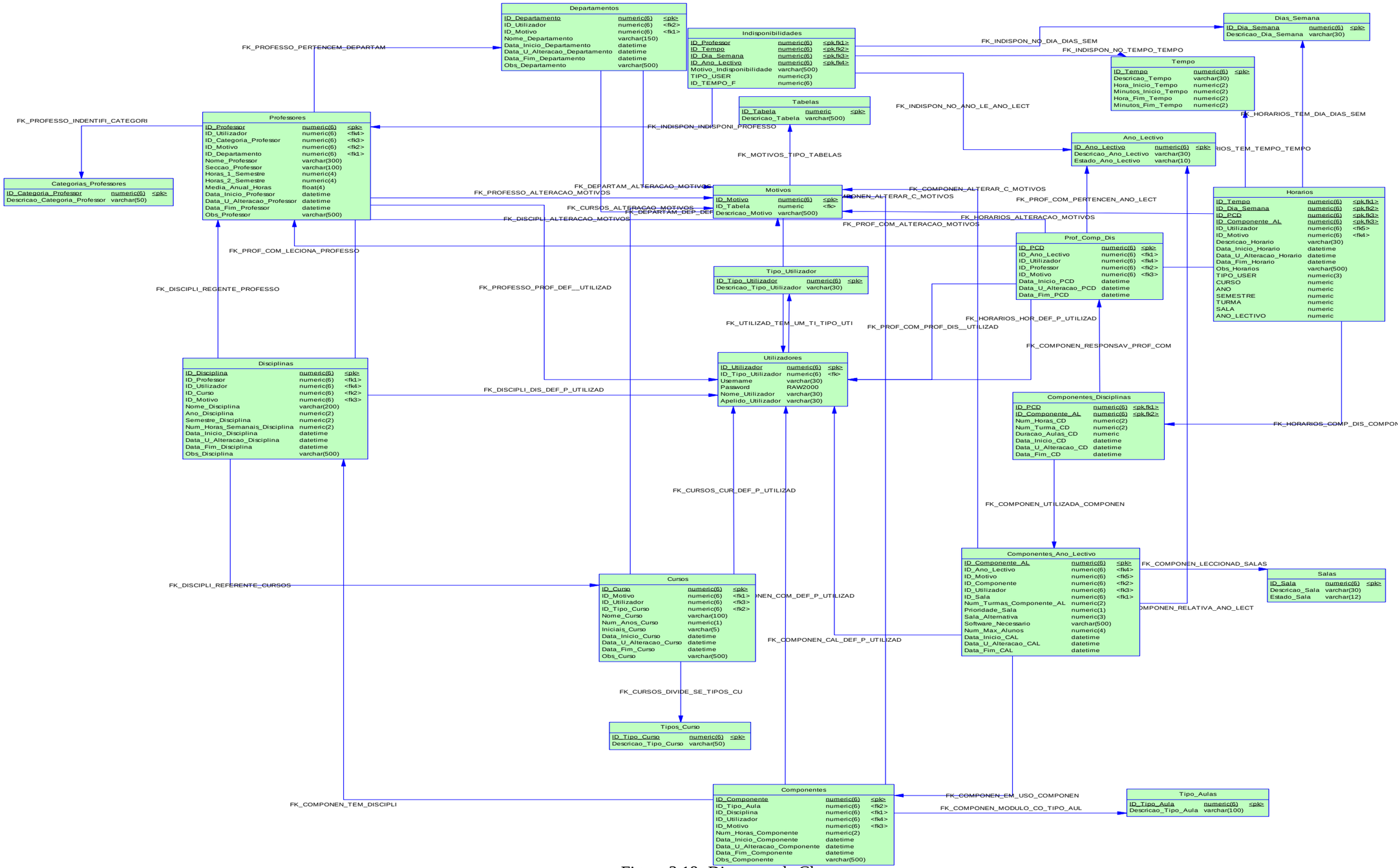

Figura 3.18- Diagrama de Classes

3.4.1. Descrição das Classes

Aqui serão descritas, permenorizadamente as classes que constituem as tabelas existentes nas bases de dados pretendendo-se então especificar todos os atributos associados as mesmas.

A descrição de cada classe irá focar quatro pontos:

1. **Atributos**: Característica que as classes possuem e que é representada por um valor de dados;

1. **Descrição**: Descrição do atributo

2. **Tipo de Dados**: Os atributos também podem ser identificados pelos seus tipo de dados, que caracteriza a informação que irá conter;

3. **Operações**: São a representação lógica do comportamento de uma classe, consistindo em acções efectuadas por ou sobre uma classe.

3.4.1.1. Classe Departamentos

Esta classe irá permitir armazenar os dados relacionados com os departamentos existentes. Todos os docentes Normalmente, os departamentos agrupam docentes que leccionam as mesmas ou áreas semelhantes. Este facto aplica-se mesmo aos docentes convidados a leccionar na instituição.

De seguida apresenta-se a descrição da classe “Departamentos” encontram-se em baixo, na tabela. na tabela é possível observar-se as suas operações.

Tabela 3.17- Descrição da Classe "Departamentos"

Atributo	Descrição	Tipo de Dados
ID_Departamento	Código que identifica a tabela Departamento (Chave Primária).	Number (10)
Nome_Departamento	Campo identificador do nome do Departamento.	Varchar2 (150)
Data_Inicio_Departamento	Data de criação do departamento.	Date
Data_U_Alteracao_Departamento	Data da última alteração efectuada no registo.	Date
Data_Fim_Departamento	Data em que o departamento deixou de ser utilizado e passou a fazer parte do histórico.	Date
Obs_Departamento	Campo onde podem ser guardadas diversas informações sobre um determinado departamento.	Varchar2 (500)

Tabela 3.18- Tabela identificadora das operações da Classe “Departamentos”

Operação	Descrição
INSERE_DEPARTAMENTOS	Operação responsável pela inserção de um novo departamento.
ALTERA_DEPARTAMENTOS	Operação responsável pela modificação de um departamento existente.
DELPROCEDURE	Operação que permite a eliminação de um determinado departamento.
CHECKIDDEPT	Operação que permite seleccionar todos os departamentos.
OBTEM_DEPARTAMENTOS	Operação que permite obter dados de um determinado Departamento.
OBTEM_DEPARTAMENTOS2	Operação que permite obter dados de todos os departamentos.

3.4.1.2. Classe Professores

Nesta tabela é armazenada toda a informação sobre os decentes que actuam e/ou actuaram na instituição.

Nas tabelas seguintes podem observar-se a descrição detalhada da classe "Professores" e as suas operações.

Tabela 3.19- Descrição da Classe "Professores"

Atributo	Descrição	Tipo de Dados
ID_Professor	Código que identifica um Professor (Chave Primária).	Number (10)
Nome_Professor	Campo identificador do nome do professor.	Varchar2 (300)
Descricao_Categoria	Campo que identifica a categoria do professor.	Varchar (100)
Seccao_Professor	Campo que identifica a secção que o professor pertence, isto se, existirem secções dentro de um departamento.	Varchar (100)
Horas_1_Semestre	Campo que identifica o número de horas que o professor lecciona por semana no 1º Semestre.	Number (2)
Horas_2_Semestre	Campo que identifica o número de horas que o professor lecciona por semana no 2º Semestre.	Number (2)
Media_Anual	Media anual do número de horas leccionadas por semana.	Float (4)
Data_Inicio_Professor	Data de criação do	Date

	professor.	
Data_U_Alteracao_ Professor	Data da última alteração efectuada no registo.	Date
Data_Fim_ Professor	Data em que o professor deixou de ser utilizado e passou a fazer parte do histórico.	Date
Obs_Professor	Campo onde podem ser guardadas variadas informações sobre um determinado professor.	Varchar2 (500)

Tabela 3.20- Tabela identificadora das operações da Classe "Professores"

Operação	**Descrição**
INSERE_PROFESSOR	Operação responsável pela inserção de um novo professor.
ALTERA_PROFESSOR	Operação responsável pela modificação de um professor existente.
DELPROCEDURE	Operação que permite a eliminação de um determinado professor.
OBTEM_PROFESSOR	Operação que permite seleccionar todos os atributos de um determinado professor.
OBTEM_PROFESSOR2	Operação que permite seleccionar todos os professores.

3.4.1.3. Classe Indisponibilidades

Na classe "Indisponibilidades" são armazenadas as indisponibilidades de cada professor para o ano lectivo corrente. As indisponibilidades de cada professor resumem-se a determinadas alturas em que um determinado docente não tem a disponibilidade para leccionar uma aula. As indisponibilidades podem englobar quaisquer razões que definam a situação acima descrita.

Tabela 3.21- Descrição da Classe "Indisponibilidades"

Atributo	Descrição	Tipo de Dados
Motivo_Indisponibilidade	Campo onde será armazenado o motivo da indisponibilidade do professor para determinado tempo e dia da semana.	Varchar2 (500)
Id_tempo_f	Campo onde será armazenado o final da indisponibilidade para aquele dia	Numeric (6,0)

Tabela 3.22- Tabela identificadora das operações da Classe "indisponibilidades"

Operação	Descrição
INSERE_INDISPONIBILIDADE	Operação responsável pela inserção de uma nova indisponibilidade do professor.
DELPROCEDURE	Operação responsável pela eliminação de uma ou mais indisponibilidades de um determinado professor.
OBTEM_ INDISPONIBILIDADE	Operação responsável por mostrar as indisponibilidades de um determinado professor.
CHECK_ID_ INDISPONIBILIDADE	Operação responsável por verificar se o numero de indisponibilidades existentes.

3.4.1.4. Classe Tipo_Utilizador

Esta classe é uma classe estática, pois os seus dados não poderão ser alterados. Irá armazenar os tipos de utilizador existentes no sistema (Administrador e Funcionário).

Tabela 3.23- Descrição da Classe "Tipo_Utilizador"

Atributo	Descrição	Tipo de Dados
ID_Tipo_Utilizador	Campo que identifica um tipo de utilizador (chave primária).	Number (1)
Descricao_Tipo_Utilizador	Campo que identifica o nome do tipo de utilizador.	Varchar2 (30)

Tabela 3.24 - Tabela identificadora das operações da Classe "Tipo_Utilizador"

Operação	Descrição
OBTEM_TIPO_UTILIZADOR	Operação responsável por seleccionar os dados de um determinado utilizador
OBTEM_TIPO_UTILIZADOR2	Operação responsável por seleccionar todos os tipos de utilizador permitindo assim a sua visualização.

3.4.1.5. Classe Utilizadores

A Classe "Utilizadores" contém todos os utilizadores da aplicação. A classe não serve apenas para validar a entrada de utilizadores na aplicação, serve também para associar os utilizadores às inserções, alterações e eliminações que efectuam aos registos da BD.

Tabela 3.25- Descrição da classe "Utilizadores"

Atributo	Descrição	Tipo de Dados
ID_Utilizador	Campo que identifica um utilizador (chave primária).	Number (10)
Username	Campo que identifica o *Username* do utilizador.	Varchar2 (30)
Password	Campo que identifica a *Password* do utilizador.	Numeric
Nome_Utilizador	Campo que identifica o Nome do utilizador.	Varchar2 (30)
Apelido_Utilizador	Campo que identifica o Apelido do utilizador.	Varchar2 (30)
Estado_Utilizador	Campo que identifica o estado do utilizador (0 para novo utilizador, 1 para utilizador em uso, 2 para utilizador desactivado).	Number (1)

Tabela 3.26- Tabela identificadora das operações da classe "Utilizadores"

Operação	Descrição
INSERE_UTILIZADOR	Operação responsável pela inserção de novos utilizadores na BD.
ALTERA_UTILIZADOR	Operação responsável pela modificação dos dados de um utilizador.
OBTEM_UTILIZADOR	Operação responsável pela selecção de um determinado utilizador.

3.4.1.6. Classe Tabelas

Esta émais uma classe onde a manipulação de dados não é possível. Armazena os tipos de dados existentes aos quais se aplicam os motivos existentes para justificar a sua eliminação.

Tabela 3.27- Descrição da classe "Tabelas"

Atributo	Descrição	Tipo de Dados
ID_Tabela	Campo que identifica uma tabela (chave primária).	Number (1)
Descrição_Tabela	Campo que identifica o nome da Tabela.	Varchar2 (500)

Tabela 3.28- Tabela identificadora das operações da classe "Tabelas"

Operação	Descrição
OBTEM_TABELAS	Operação responsável pela selecção de determinada tabela
OBTEM_TABELAS2	Operação responsável pela selecção de todas as tabelas existentes, às quais se aplicam motivos.

3.4.1.7. Classe Motivos

Esta classe armazena dados referentes a motivos que levam a determinados registos a entrarem no histórico do sistema.

Tabela 3.29- Descrição da classe "Motivos"

Atributo	Descrição	Tipo de Dados
ID_Motivo	Campo que identifica um motivo (chave primária).	Number (10)
Descrição_Motivo	Campo utilizado para descrever um Motivo.	Varchar2 (500)

Tabela 3.30- Tabela identificadora das operações da classe "Motivos"

Operação	Descrição
INSERE_MOTIVOS	Operação responsável pela inserção de novos motivos.
ALTERA_MOTIVOS	Operação responsável pela alteração de motivos já existentes.
DELPROCEDURE	Operação responsável pela eliminação de motivos já existentes.
OBTEM_MOTIVOS	Operação responsável pela selecção De um determinado motivo.
OBTEM_MOTIVOS2	Operação responsável pela selecção dos motivos existentes.

3.4.1.8. Classe Disciplinas

Os dados nesta tabela são de alta importância uma vez que traduzem a informação das disciplinas leccionadas na aplicação. Através destes dados poderão obter-se informações para a distribuição das componentes que lhes estão associadas.

Poderão existir disciplinas com o mesmo nome, no entanto, essa situação não se poderá verificar dentro do mesmo curso.

Tabela 3.31- Descrição da classe "Disciplinas"

Atributo	Descrição	Tipo de Dados
ID_Disciplina	Campo que identifica uma Disciplina (Chave Primária).	Number (10)
Nome_Disciplina	Campo que identifica o nome de uma Disciplina.	Varchar2 (200)
Ano_Disciplina	Campo que identifica o ano de uma Disciplina.	Number (2)
Semestre_Disciplina	Campo que identifica o semestre de uma Disciplina.	Varchar2 (1)
Num_Horas_Semanais_Disciplina	Campo que identifica o número de horas semanais de uma disciplina.	Number (2)
Data_Inicio_Disciplina	Campo que identifica a data de inserção da Disciplina.	Date
Data_U_Alteracao_Disciplina	Campo que identifica a data da ultima alteração na disciplina.	Date
Data_Fim_Disciplina	Campo que identifica a data de fim de utilização de uma disciplina.	Date
Obs_Disciplina	Campo onde podem ser guardadas diversas informações sobre uma determinada Disciplina.	Varchar2 (500)

Tabela 3.32- Tabela identificadora das operações da classe "Disciplinas"

Operação	Descrição
INSERE_DISCIPLINAS	Operação responsável pela inserção de novas disciplinas.
ALTERA_DISCIPLINAS	Operação responsável pela alteração de disciplinas já existentes.
CHECKID_DISCI	Operação responsável pela verificação do número total de disciplinas.
DELPROCEDURE	Operação responsável pela eliminação de uma disciplina.
OBTEM_DISCIPLINA	Operação responsável pela selecção dos atributos de uma determinada disciplina.
OBTEM_DISCIPLINAS2	Operação responsável pela selecção das disciplinas existentes.
OBTEM_DISCIPLINA_PROF	Operação responsável pela selecção dos atributos das disciplinas associadas a um determinado docente
OBTEM_DISCIPLINA_DEPT	Operação responsável pela selecção dos atributos das disciplinas associadas a um determinado departamento.
OBTEM_DISCIPLINA_SEME	Operação responsável pela selecção dos atributos das disciplinas associadas a um determinado semestre
OBTEM_DISCIPLINA_CURS	Operação responsável pela selecção dos atributos das disciplinas associadas a um determinado curso

3.4.1.9. Classe Cursos

Na classe "Cursos" é guardada toda a informação relativa aos cursos existentes na instituição. Também esta classe armazena dados importantes.

Tabela 3.33- Descrição da classe "Cursos"

Atributo	Descrição	Tipo de Dados
ID_Curso	Campo que identifica um curso (Chave Primária).	Number (10)
Nome_Curso	Campo que identifica o nome de um curso.	Varchar2 (100)
Num_Anos_Curso	Campo que identifica o número de anos de um curso.	Number (1)
Iniciais_Curso	Campo que representa a iniciais do curso.	Varchar2 (5)
Data_Inicio_Curso	Campo que identifica a data de inserção do curso.	Date
Data_U_Alteracao_ Curso	Campo que identifica a data da ultima alteração no curso.	Date
Data_Fim_Curso	Campo que identifica a data de fim de utilização do curso.	Date
Obs_Curso	Campo onde podem ser guardadas variadas informações sobre um determinado Curso.	Varchar2 (500)

Tabela 3.34- Tabela identificadora das operações da classe ”Cursos”

Operação	Descrição
INSERE_CURSO	Operação responsável pela inserção de um novo curso.
ALTERA_CURSO	Operação responsável pela alteração de um curso já existente.
CHECKID_CURSO	Operação responsável pela verificação do numero total de cursos
DELPROCEDURE	Operação responsável pela eliminação de um curso.
OBTEM_CURSO	Operação responsável pela selecção de um determinado curso
OBTEM_CURSO2	Operação responsável pela selecção dos cursos existentes.

3.4.1.9. Classe Tipos_Curso

Nesta classe serão armazenados os dados referentes aos tipo de curso leccionados na instituição

Tabela 3.35- Descrição da classe

"Tipos_Curso" Atributo	Descrição	Tipo de Dados
ID_Tipo_Curso	Campo responsável pela identificação de um tipo de curso (chave primária).	Number (10)
Descricao_Tipo_Curso	Campo responsável pela descrição de um tipo de curso.	Varchar2 (50)

Tabela 3.36- Tabela identificadora das operações da classe "Tipos_Curso"

Operação	Descrição
INSERE_TIPO_CURSO	Operação responsável pela inserção de novo tipo de curso.
ALTERA_TIPO_CURSO	Operação responsável pela alteração de um tipo de curso já existente.
DELPROCEDURE	Operação responsável pela eliminação de um tipo de curso.
OBTEM_ TIPO_CURSO	Operação responsável pela selecção um determinado tipo de curso.
OBTEM_ TIPO_CURSO2	Operação responsável pela selecção dos tipos de curso existentes.

3.4.1.10. Classe Componentes

As aulas são leccionadas através de componentes que são nada mais e nada menos que tipos de aula. Esta classe alberga a informação relacionada com a disposição das componentes das disciplinas

Tabela 3.37- Descrição da classe "Componentes"

Atributo	Descrição	Tipo de Dados
ID_Componente	Campo responsável pela identificação de uma componente (chave primária).	Number (10)
Num_Horas_Componente	Campo responsável pelo número de horas semanais de uma determinada componente.	Number (2)
Data_Incio_Componente	Data de criação da componente.	Date
Data_U_Alteracao_Componente	Data da última alteração de uma determinada componente.	Date
Data_Fim_Componente	Data de eliminação de um determinada componente.	Date
Obs_Componente	Campo onde podem ser guardadas variadas informações sobre uma determinada componente.	VarChar2 (500)

Tabela 3.38- Tabela identificadora das operações da classe "Componentes"

Operação	Descrição
INSERE_COMPONENTES	Operação responsável pela inserção de uma nova componente
ALTERA_COMPONENTES	Operação responsável pela alteração de uma componente
OBTEM_COMPONENTES	Operação responsável pela visualização das componentes existentes
CHECKID_COMPONENTES	Operação responsável pela selecção do número total de componentes
OBTEM_COMPONENTES2	Operação responsável pela selecção do total das componentes.
DELPROCEDURE	Operação responsável pela eliminação de uma determinada componente

3.4.1.12. Classe Tipo_Aulas

As aulas são leccionadas em diversos tipos de aula, por exemplo: aulas teóricas , praticas ou ainda teórico-práticas. É pertinente então que os dados referentes a estes tipos de aula sejam armazenados para que as componentes das aulas possam ser documentadas convenientemente.

Tabela 3.39- Descrição da classe "Tipo_Aulas"

Atributo	Descrição	Tipo de Dados
ID_Tipo_Aula	Campo responsável pela identificação de um tipo de Aula (chave primária).	Number (10)
Descricao_Tipo_Aula	Campo responsável pela descrição de um tipo de aula.	Varchar2 (100)

Tabela 3.40- Tabela identificadora das operações da classe "Tipo_Aulas"

Operação	Descrição
INSERE_TIPO_AULAS	Operação responsável pela inserção de novo tipo de aula.
ALTERA_TIPO_AULAS	Operação responsável pela alteração de um tipo de aula já existente.
CHECK_TIPO_AULAS	Operação responsável pelo calculo do total de tipos de aula existentes.
DELPROCEDURE	Operação responsável pela eliminação de um tipo de aula.
OBTEM_TIPO_AULAS	Operação responsável pela selecção dos tipos de aulas existentes.
OBTEM_TIPO_AULAS2	Operação responsável pela selecção do total de tipos de aula existenes

3.4.1.13. Classe Componentes_Ano_lectivo

Esta classe é importante porque especifica dados importantes das componentes que sofrem actualizações constantes. Os dados nesta classe passam por atributos tais como: sala em que é leccionada, número de alunos, conteúdos ou número de horas semanais.

A sua actualização é efectuada todos os anos lectivos e mesmo durante os mesmos e uma das bases para a criação de horários de cada ano lectivo.

Tabela 3.41- Descrição da classe "Componentes_Ano_Lectivo"

Atributo	Descrição	Tipo de Dados
ID_Componente_AL	Campo responsável pela identificação de uma componente no ano lectivo (chave primária).	Number (10)
Num_Turmas_Componente_AL	Campo responsável pelo número turmas que uma componente no lectivo possui.	Number (2)
Prioridade_Sala	Campo responsável pela prioridade que um sala tem para uma determinada componente (valores entre 1 e 3).	Number (1)
Sala_Alternativa	Componente que identifica a sala alternativa caso a principal não possa ser utilizada.	Varchar2 (30)
Material_Necessario	Componente que identifica o material necessário para as aulas de	Varchar2 (500)

	uma determinada componente.	
Num_Max_Alunos	Componente que identifica o número máximo de alunos que uma aula deve ter.	Number (3)
Data_Inicio_CAL	Data de inserção da componente no ano lectivo.	Date
Data_U_Alteracao_CAL	Data da última alteração na componente no ano lectivo.	Date
Data_Fim_CAL	Data de eliminação da componente no ano lectivo.	Date

Tabela 3.42- Tabela identificadora das operações da classe Componentes_Ano_Lectivo"

Operação	Descrição
INSERE_COMPONENTE_AL	Operação responsável pela inserção de nova componente no ano lectivo.
ALTERA_COMPONENTE_AL	Operação responsável pela alteração de uma componente no ano lectivo.
DELPROCEDURE	Operação responsável pela eliminação de uma componente no ano lectivo.
CHECKID_COMPONENTE_AL	Operação responsável pelo calculo do total das componentes existentes.
OBTEM_COMPONENTE_AL	Operação responsável pela selecção dos dados de uma componente no ano lectivo.
OBTEM _COMPONENTE_AL2	Operação responsável pela visualização das componentes existentes.

3.4.1.14. Classe Salas

É importante salientar que as aulas são leccionadas em salas de aula ou laboratórios. Assim, os dados referentes a todas as salas de aula serão armazenados nesta classe. Esses dados serão a sala em questão e o seu estado senod o "Operacional" o considerado para as salas de aula onde podem ser dadas aulas, "Reservado" para as salas que são ocupadas por outras entidades e "Em manutenção" para as salas que se encontram em manutenção alargadae não podem ser utilizadas de momento.

Tabela 3.43- Descrição da classe "Salas"

Atributo	Descrição	Tipo de Dados
ID_Sala	Campo que identifica uma Sala (Chave Primária).	Number (10)
Descricao_Sala	Campo que identifica o nome de uma sala.	Varchar2 (30)
Estado_Sala	Campo que identifica o estado de uma sala (disponível ou indisponível).	Varchar2 (12)

Tabela 3.44- Tabela identificadora das operações da classe "Salas"

Operação	Descrição
INSERE_SALA	Operação responsável pela inserção de nova sala.
ALTERA_SALA	Operação responsável pela alteração de uma sala.
DELPROCEDURE	Operação responsável pela eliminação de uma sala.
CHECKID_SALA	Operação responsável pelo calculo do total
OBTEM_SALA	Operação responsável por obter dados de uma determinada sala
OBTEM_SALA2	Operação responsável por obter os dados de todas as salas existentes.

3.4.1.15. Classe Componentes_Disciplinas

Esta classe reúne todos os dados referentes às disciplinas que são leccionadas a cada turma bem como quem as lecciona e a duração das aulas. Esta é a fonte de onde a classe horários irá retirar os dados referentes os aulas. A chave primaria desta classe será constituída pelos atributos estrangeiros: ID_Componente_AL de **"**Componentes_Ano_Lectivo**" e** ID_PCD de **"**Componentes_Disciplina**"**

Tabela 3.45- Descrição da classe "Componentes_Disciplinas"

Atributo	Descrição	Tipo de Dados
Num_Horas_CD	Campo que identifica o número de horas da componente.	Number (2)
Num_Turma_CD	Campo que identifica o número da turma da componente.	Number (2)
Duracao_Aulas_CD	Campo que identifica a duração estipulada das aulas.	Number (4)
Data_Inicio_CD	Data em que a componente foi criada.	Date
Data_U_Alteracao_CD	Data da última alteração efectuada ao registo.	Date
Data_Fim_CD	Data em que o registo foi "eliminado".	Date

Tabela 3.46- Tabela identificadora das operações da classe "Componentes_Disciplina"

Operação	Descrição
INSERE_COPMDIS	Operação responsável pela inserção de uma nova componente.
ALTERA_COPMDIS	Operação responsável pela alteração de uma componente existente.
DELPROCEDURE	Operação responsável pela eliminação de uma componente.
CHECKID_COPMDIS	Operação responsável pelo calculo do total de registos existentes na classe
OBTEM_COPMDIS	Operação responsável por listar os dados de um determinado registo
OBTEM _COPMDIS2	Operação responsável por todos os registos da classe.

3.4.1.16. Classe Prof_Com_Dis

Tendo em conta que a classe anterior pode ser considerada a mais importante no que toca à inserção de dados na classe Horários. Esta é sem duvida uma das classes mais importantes para a inserção de dados na classe anterior. Uma vez que dois docentes podem dar aulas, a mesma componente e a mesma disciplina, é conveniente então diferenciar ambos os casos como casos singulares.

Tabela 3.47- Descrição da classe ”Prof_Comp_Disc”

Atributo	Descrição	Tipo de Dados
ID_PCD	Campo que identifica o Prof_Comp_Dis (chave primária).	Number (10)
Data_Inicio_PCD	Data em que o registo foi criado.	Date
Data_U_Alteracao_PCD	Data da última alteração efectuada ao registo.	Date
Data_Fim_PCD	Data em que o registo foi “eliminado”.	Date

Tabela 3.48- Tabela identificadora das operações da classe “Prof_Comp_Disc”

Operação	Descrição
INSERE_PROF_COMP_DIS	Operação responsável pela inserção de um novo Prof_Comp_Disc.
ALTERA_PROF_COMP_DIS	Operação responsável pela alteração de um Prof_Comp_Disc.
DELPROCEDURE	Operação responsável pela eliminação de um Prof_Comp_Disc.
CHECKID_PROF_COMP_DIS	Operação responsável pelo calculo do total de registos existentes
OBTEM_PROF_COMP_DIS	Operação responsável pela obtenção de dados de um determinado registo
OBTEM_PROF_COMP_DIS2	Operação responsável pela obtenção de todos os registos

3.4.1.17. Classe Horarios

Pode dizer-se que esta classe é o centro da aplicação para a qual todas a outras têm significado directo ou indirecto. A base de dados foi então desenhada tendo esta classe como a mais importante e para a mesma que a aplicação se desenvolve.

Tabela 3.49- Descrição da classe "Horário"

Atributo	Descrição	Tipo de Dados
Curso	Campo que identifica o curso do horário.	Number (2)
Ano	Campo que identifica o ano do horário.	Number (2)
Semestre	Campo que identifica o semestre do horário.	Number (1)
Turma	Campo que identifica a turma do horário.	Number (2)
Sala	Campo que identifica a sala de aula onde a aula será leccionada	Number (18)
Ano_Lectivo	Campo que identifica o ano lectivo aplicado	Number (18)

Tabela 3.50- Tabela identificadora das operações da classe "Horário"

Operação	Descrição
INSERE_HORARIO	Operação responsável pela inserção de um novo horário.
ALTERA_HORARIO	Operação responsável pela alteração de um horário.
DELPROCEDURE	Operação responsável pela eliminação de um horário.
OBTEM _HORARIO	Operação responsável pela visualização de um horário.

3.4.1.18. Classe Dias_Semana

Mais uma classe estática cujos dados são inalteráveis servindo de apoio à inserção de dados em outras diferentes classes. É estatica uma vez que a semana tem apenas 7 dias ☺

Tabela 3.51- Descrição da classe "Dias_Semana"

Atributo	Descrição	Tipo de Dados
ID_Dias_Semana	Campo que identifica um dia da semana (chave primária).	Number (1)
Descrição_Dia_Semana	Campo que define a descrição de um dia da semana.	Varchar2 (30)

Tabela 3.52- Tabela identificadora das operações da classe "Dias_Semana"

Operação	Descrição
OBTEM_DIA_SEMANA	Operação responsável pela obtenção de um determinado dia da semana
OBTEM_DIA_SEMANA2	Operação responsável pela selecção e obtenção de todos os registos

3.4.1.19 Classe Tempo

Esta classe estática armazena registos de horas as quais são atribuídas as aulas. Cada intervalo tem uma duração de 50 minutos que vão das 08h30m até as 23h20m.

Tabela 3.53– Descrição da tabela “Tempo

Atributo	Descrição	Tipo de Dados
ID_Tempo	Campo que identifica um tempo (chave primária).	Number (2)
Descrição_Tempo	Campo que define a descrição de um tempo.	Varchar2 (30)
Hora_Inicio_Tempo	Campo que define a hora de início de um tempo.	Number (2)
Minutos_Inicio_Tempo	Campo que define os minutos de início de um tempo.	Number (2)
Hora_Fim_Tempo	Campo que define a hora de fim de um tempo.	Number (2)
Minutos_Fim_Tempo	Campo que define os minutos de fim de um tempo.	Number (2)

Tabela 3.54- Tabela identificadora das operações da tabela "Tempo"

Operação	Descrição
OBTEM_ID_TEMPO	Operação responsável pela selecção de um determinado registo
OBTEM_ID_TEMPO2	Operação responsável pela obtenção de todos os dados existentes.

3.4.1.20. Classe Ano_lectivo

Como muita informação no sistema depende do ano lectivo onde é utilizada deixando de ter sentido quando se muda de ano lectivo. Uns dos exemplos óbvios são mesmo os horários que mudam todos os anos e as informações associadas perdem o sentido.

Tabela 3.55- Descrição da classe "Ano_Lectivo"

Atributo	Descrição	Tipo de Dados
ID_Ano_Lectivo	Campo que identifica um ano lectivo (chave primária).	Number (6)
Descrição_Ano_Lectivo	Campo que define a descrição de um ano lectivo.	Varchar2 (30)
Estado_Ano_Lectivo	Campo que define o estado de um ano lectivo, em uso ou não.	Number (1)

Tabela 3.56- Tabela identificadora das operações da classe "Ano_Lectivo"

Operação	Descrição
INSERE_ANO_LECTIVO	Operação responsável pela inserção de um novo ano lectivo.
ALTERA_ANO_LECTIVO	Operação responsável pela alteração do ano lectivo actual.
DELPROCEDURE	Operação responsável pela eliminação do ano lectivo actual
OBTEM_ANO_LECTIVO	Operação responsável pela obtenção de um determinado ano lectivo
OBTEM_ANO_LECTIVO2	Operação responsável pela obtenção de todos os registos existentes.

4. Implementação

Aqui termina a análise do sistema, a qual foi executada em conjunto com o aluno Manuel Vieira. Começa então a implementação do sistema que será documentada o mais clara possível.

Assim a implementação será documentada seguindo os passos:

4.1. Modelo de Entidades Relacional (ER)

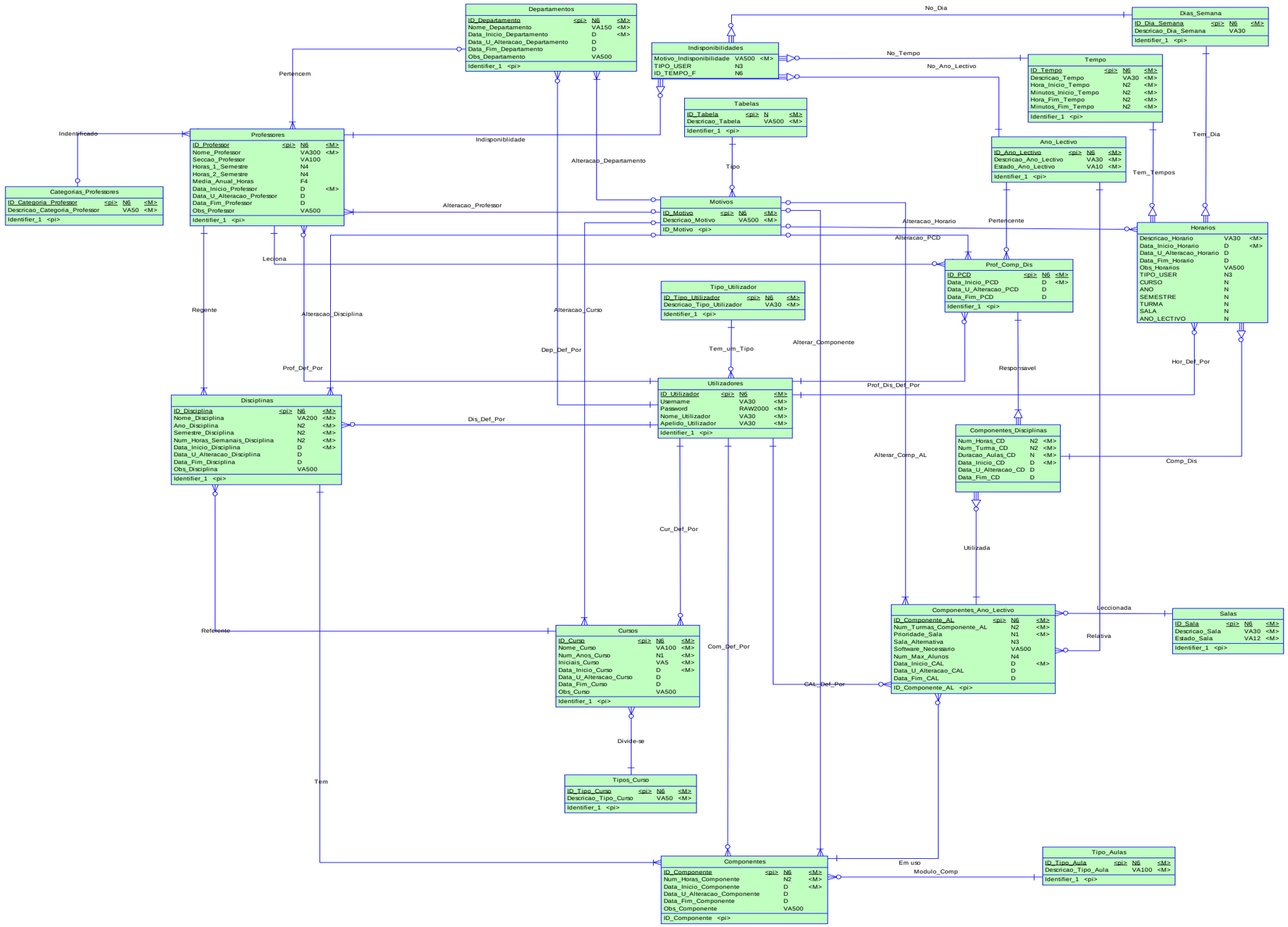

Figura 4.1– Modelo Relacional

Para a implementação desta aplicação, inicialmente, procedeu-se ao levantamento de informação a qual mais tarde seria processada para iniciar o desenvolvimento.

Para começar, houve uma reunião com a funcionária Cristina Rosa, a qual, explicou o procedimento e os documentos utilizados para a produção de horários.

Tendo isto, começaram as reuniões com o Professor José Fonseca e o meu colega Manuel Vieira de forma a transformar as primeiras 6 tabelas deram origem ao modelo ER final que visa dar conta de todas as eventualidades e processos para a criação de horários.

Após várias reuniões e mais tarde depois da análise do sistema, iniciou-se a implementação da aplicação. O meu colega partiu para a produção do sistema numa aplicação Windows criada em C# com e eu parti para o desenvolvimento de uma aplicação Web em JSP.

O modelo ER foi produzido em o *PowerDesigner* da *Sybase* v12 o qual permite a criação fácil e simples de modelos conceptuais, físicos, de objectos . Permite ainda a criação de ficheiros DDL para a criação das bases de dados em MS SQL Server 2005 e em Oracle 10G R2.

4.2. Criação das Bases de Dados

Assim procedeu-se à criação das bases de dados.

Após a criação do modelo conceptual, que equivale ao modelo ER, devemos criar o modelo físico ou diagrama de classes.

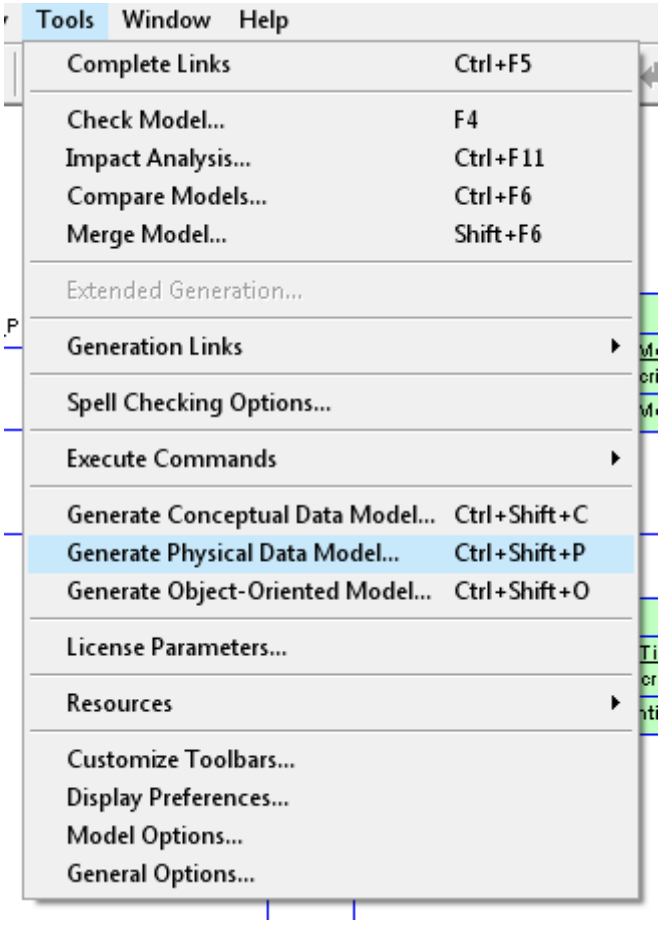

Figura 4.2– Acesso à criação do Modelo Físico (Diagrama de Classes)

De seguida:

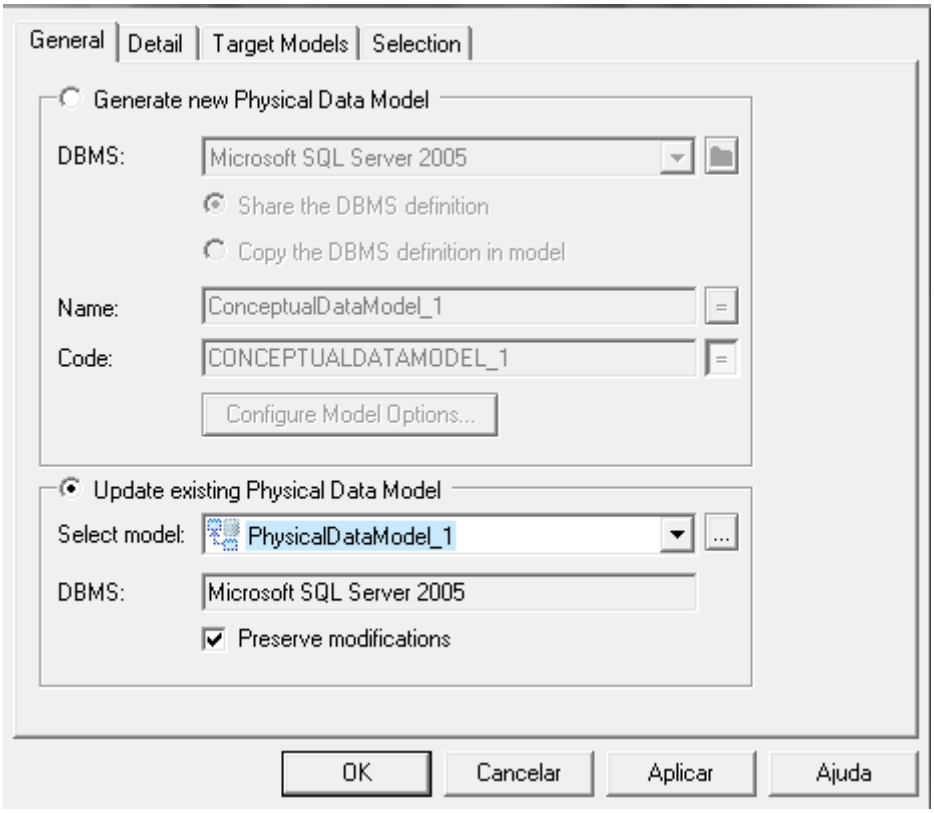

Figura 4.3 - Criação do Modelo Físico (Diagrama de Classes)

Depois procedeu-se à exportação para os diferentes Gestores de Bases de Dados

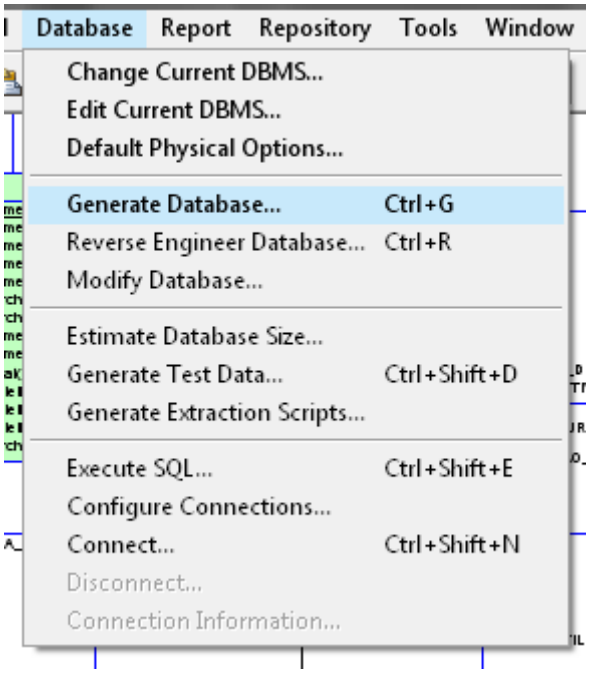

Figura 4.4– Acesso ao menu de criação de bases de dados

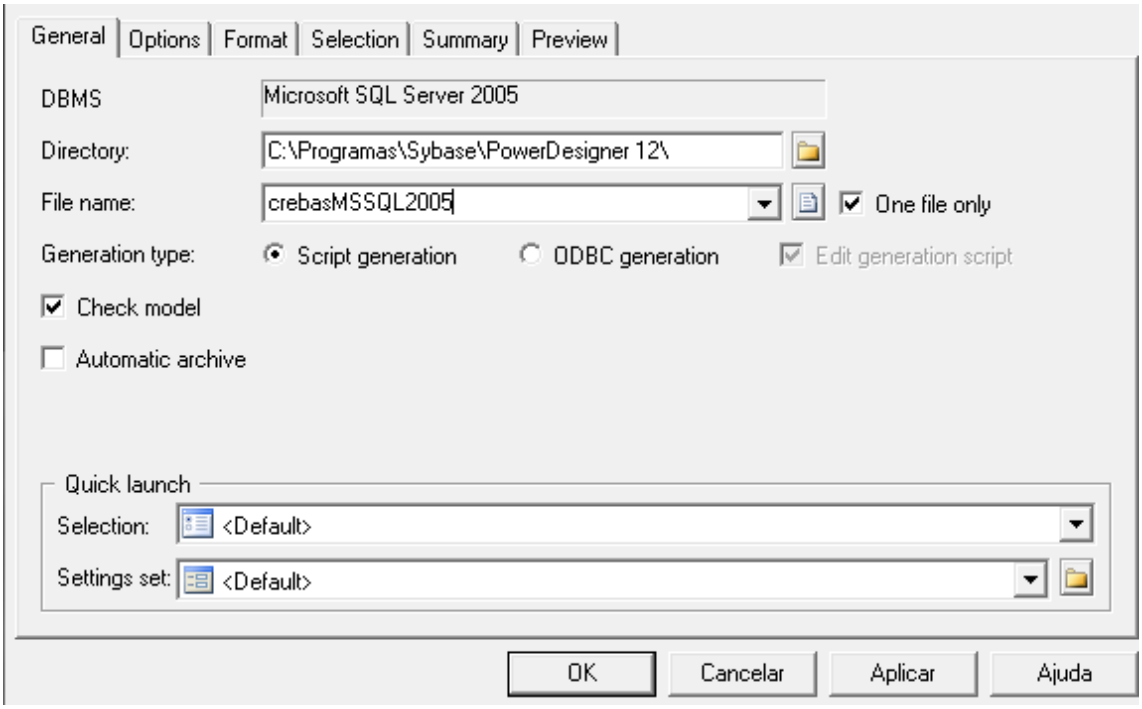

Figura 4.5– Menu de criação de bases de dados para SQL Server

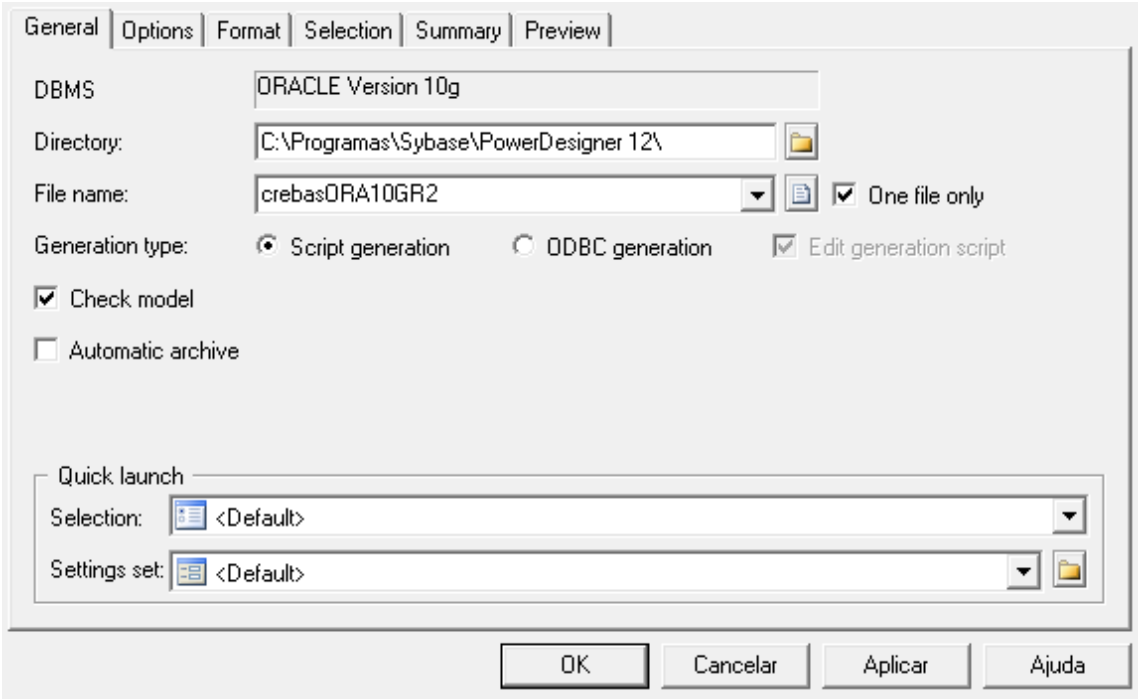

Figura 4.6 – Menu de criação de bases de dados para Oracle

4.3. T-SQL e PL/SQL

Após a criação das bases de dados procede-se a programação de procedimentos que irão interagir com as bases de dados.

Aqui esta a base de um procedimento criado em T-SQL.

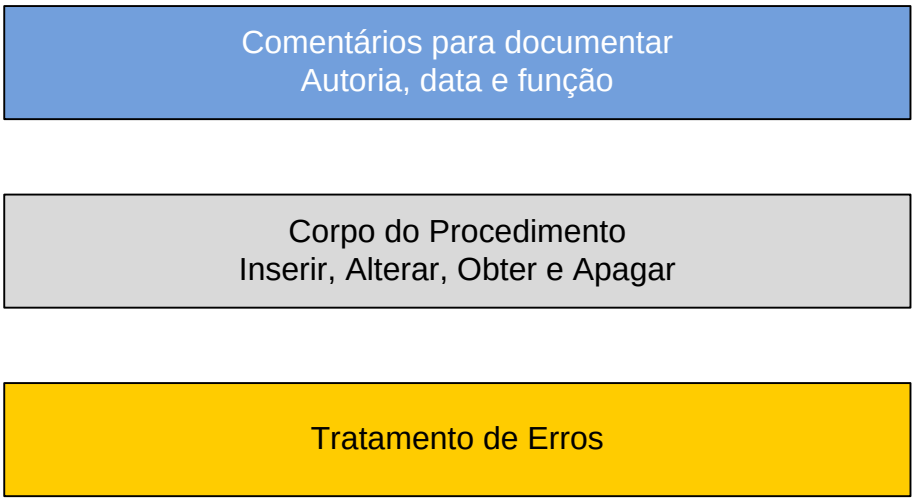

Figura 4.7– Arquitectura base de um procedimento em T-SQL

Aqui está um procedimento base em PL/SQL

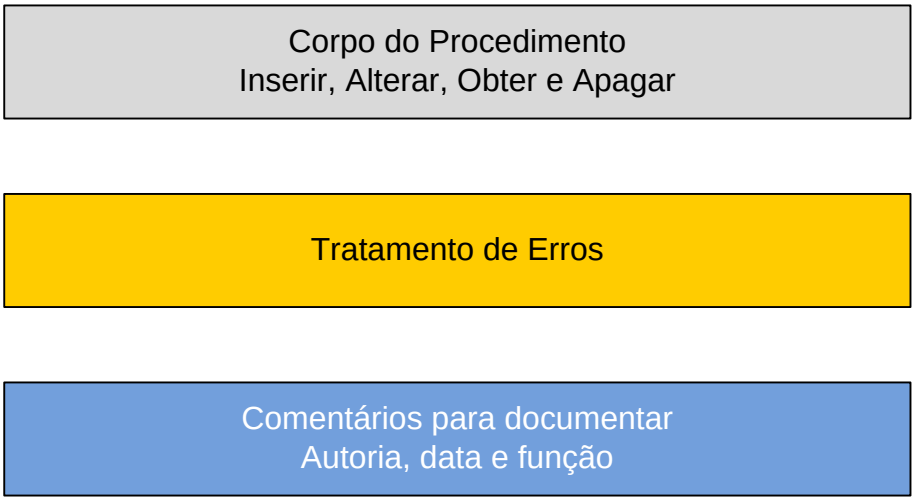

Figura 4.8– Arquitectura base de um procedimento em PL/SQL

Nos anexos serão apresentados procedimentos utilizados pelo sistema

4.4. Acessos

O acesso às bases de dados é efectuado através de **Java Database Connectivity** (JDBC) a qual é feita através de *drivers* específicos para cada gestor de bases de dados.

Neste caso, usou-se o *driver* 1.4 para conexão Oracle e a versão 1.1 para a conexão com o MS SQL Server.

Tendo isto a conexão é feita essencialmente com três linhas de código:

```
Class.forName(drivermssql);
conn_insINDITSQL = DriverManager.getConnection(constrmssql);
CallableStatement cs_insCATTSQL = conn_insINDITSQL.prepareCall("{call dbo.OBTEM_INDISPONIBILIDADES2(?)}");
```

Figura 4.9– Código para conexão às bases de dados

Onde a verde "drivermssql" é identificado o driver utilizado. A frase de conexão (*connection string*) encontra-se em "constrmssql" onde os dados do servidor ou computador onde se encontram as bases de dados, do utilizador, a palavra-passe de acesso e a porta de conexão. As frases de conexão variam conforme o gestor de bases de dados

Como norma de segurança, os dados referentes as conexões encontram-se armazenadas num ficheiro XML armazenado no disco.

Os dados são acedidos através de JavaBeans quer aos dados do ficheiro XML quer as bases de dados. Estas classes servem como intermediarias entre a interface (JSP) e as bases de dados.

4.5. JavaBeans

Como dito anteriormente, estas classes são os intermediários que comunicam entre a interface e as bases de dados.

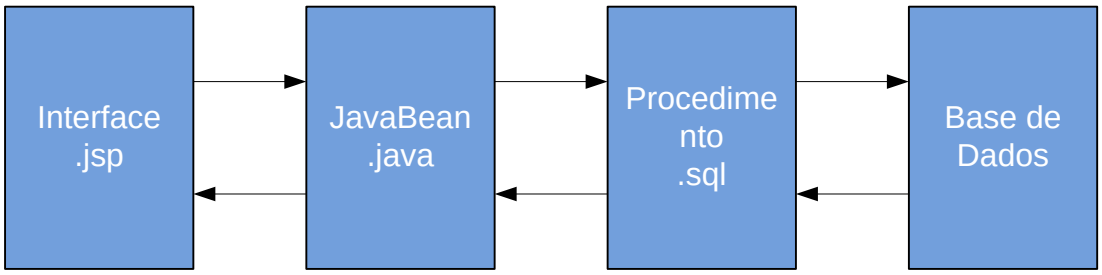

Figura 4.10– Interacção entre entidades.

São nestas classes, criadas funções para a interacção com as bases de dados onde dados são, ou não, inseridos dados vindos da interface e estas funções devolvem a informação requerida, seja por operações ou por tratamentos de dados.

Foram criadas classes para operações e para acessos as bases de dados. As referentes a conexões à base de dados contêm uma função específica para cada acção em cada base de dados.

```
public void setHostString(String[] inhostdata) {...}
public String[] CHECKIDCURS_MSSQL(int idtypeuser) {...}
public String[] CHECKIDCURS_ORASQL(int idtypeuser) {...}
public String[] INSCURS_MSSQL(int idtypeuser, String[] indept) {...}
public String[] INSCURS_ORASQL(int idtypeuser, String[] indept) {...}
public String[] ALTCURS_MSSQL(int idtypeuser, String[] indept) {...}
public String[] ALTCURS_ORASQL(int idtypeuser, String[] indept) {...}
public String[] OBTNUMCURS_MSSQL(int idtypeuser) {...}
public String[] OBTNUMCURS_ORASQL(int idtypeuser) {...}
public String[] OBTCURS_MSSQL(int idtypeuser, String[] indept) {...}
public String[] OBTCURS_ORASQL(int idtypeuser, String[] indept) {...}
public String[] getCURSO_MSSQL(int idtypeuser) {...}
public String[] getCURSO_ORASQL(int idtypeuser) {...}
public void SETERROR_MSSQL(int ERRORNUMBER, String ERRORSEVERITY, String ERRORSTATE,
        String ERRORPROCEDURE,
        String ERRORLINE,
        String ERRORMESSAGE) {...}
public void SETERROR_ORASQL(int ERRORNUMBER, String ERRORMESSAGE) {...}
public void SETEXCEP_MSSQL(String ERRORMESSAGE) {...}
public void SETEXCEP_ORASQL(String ERRORMESSAGE) {...}
```

Figura 4.11– Funções internas de um JavaBean.

Apresentadas na figura anterior estão representadas a funções par a interacção com as bases de dados.

As suas funções resumem-se a:

Tabela 4.1- Descrição de funções existentes numa JavaBean"

CHECKIDCURS_XXXSQL	Função responsável pela verificação de ID's
INSCURS_XXXSQL	Função responsável pela inserção de dados
ALTCURS_XXXSQL	Função responsável pela alteração de dados
OBTNUMCURS_XXXSQL	Função responsável pela obtenção do número total de registos
OBTCURS_XXXSQL	Função responsável pela obtenção de dados de um determinado registo
getCURSO_XXXSQL	Função responsável pela obtenção de dados de todos os registos da tabela
SETERROR_XXXSQL	Função responsável pelo armazenamento temporário de erros quando existem
SETEXCEP_XXXSQL	Função responsável pelo armazenamento temporário de excepções quando existem
GETERROR_XXXSQL	Função responsável pelo envio para a interface de excepções quando existem

4.6. Java Server Pages (JSP)

Quando se iniciou a implementação desta aplicação a ideia de tornar aplicação portável foi a mais aliciante. A ideia inicial seria proceder à implementação de uma aplicação Web em ASP .NET com incorporação de C#. Contudo como o aluno Manuel Vieira decidiu utilizar esta linguagem para a sua aplicação decidiu-se passar a uma outra forma de implementar a aplicação.

Todas as paginas JSP estão estruturadas para uma função especifica normalmente relacionada com uma operação de uma determinada tabela.

4.6.1. Obtenção de todos dados

Que pode ser acedida através do menu principal ou de um *link* adjacente quando os dados se encontram associados a outras tabelas

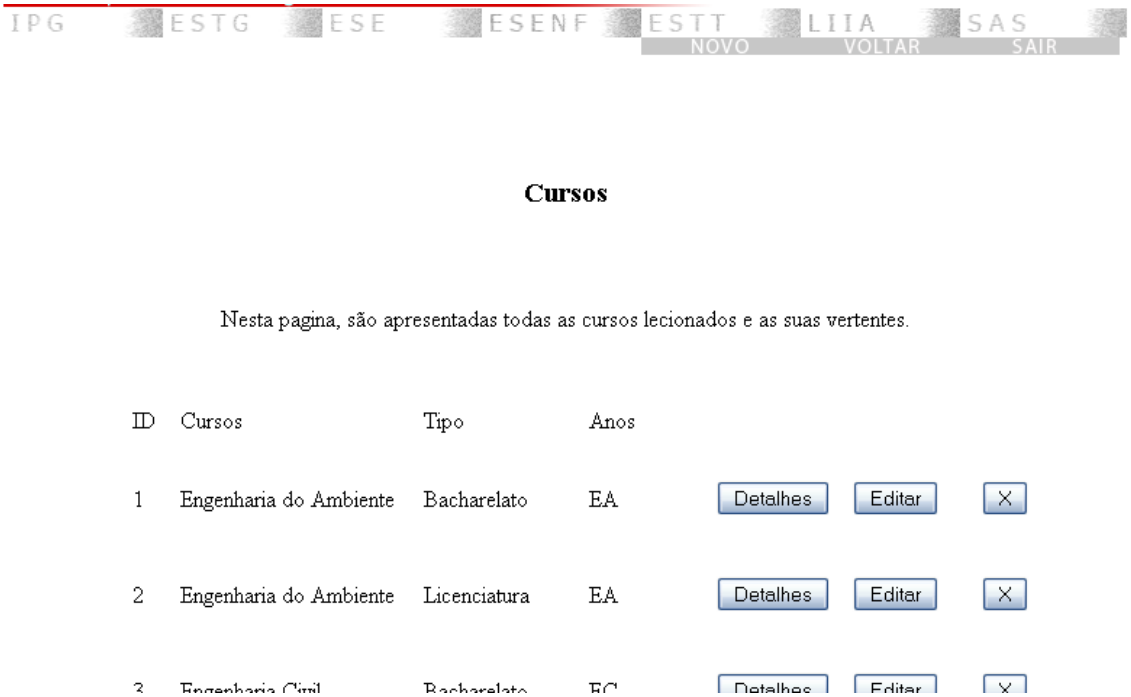

Figura 4.12– Obtenção de dados de uma tabela.

Inicialmente a pagina faz um pedido à JavaBean correspondente que usa uma função do tipo:

```
public String[] getCURSO_MSSQL(int idtypeuser) {

    dim = CHECKIDCURS_MSSQL(idtypeuser);
    int indim = Integer.valueOf(dim[1]).intValue();
    indim = (indim * 11);
    String bigarray[] = new String[indim];

    int i = 0;
    java.util.Arrays.fill(saida, " ");
    try {

        Class.forName(drivermssql);
        conn_insCURSTSQL = DriverManager.getConnection(constrmssql);
        CallableStatement cs_insCURSTSQL = conn_insCURSTSQL.prepareCall("{call dbo.OBTEM_CURSO2(?)}");

        cs_insCURSTSQL.setInt(1, idtypeuser);
        ResultSet rs = cs_insCURSTSQL.executeQuery();

        while (rs.next()) {

            bigarray[i] = rs.getString(1);
            bigarray[i + 1] = rs.getString(2);
            bigarray[i + 2] = rs.getString(3);
            bigarray[i + 3] = rs.getString(4);
            bigarray[i + 4] = rs.getString(5);
            bigarray[i + 5] = rs.getString(6);
            bigarray[i + 6] = rs.getString(7);
            bigarray[i + 7] = rs.getString(8);
            bigarray[i + 8] = rs.getString(9);
            bigarray[i + 9] = rs.getString(10);
            bigarray[i + 10] = rs.getString(11);
            i = i + 11;
        }
        rs.close();


    } catch (Exception e) {
        if (e.getMessage() != null) {

            SETEXCEP_MSSQL(e.getMessage());
        }
```

Figura 4.13– Funções internas de uma JavaBean

Que devolverá os dados de todos registos existentes. De seguida, a pagina JSP interpretará os dados de forma a mostra-los.

```
if (BD == 1) {

    dim = BEAN_CURS.CHECKIDCURS_MSSQL(bdtypeuser);
    int indim = Integer.valueOf(dim[1]).intValue();
    indim = (indim * 11) + 11;
    String indata[] = new String[indim];

    indata = BEAN_CURS.getCURSO_MSSQL(bdtypeuser);

    for (i = 0; i < indata.length; i++) {%>
    <tr>
        <% idenv = Integer.valueOf(indata[i]).intValue();
                if (indata[i] != null && indata[i] != " ") {%>
        <td height="64" align="left" valign="middle"><%out.print(indata[i]);%></td>
        <td valign="middle"><%out.print(indata[i + 4]);%></td>
        <td valign="middle"><%
        //out.print();
            inar2[0] = indata[i + 3];
            if (BD==1){
            outdata2 = BEAN_TICU.OBTTICU_MSSQL(bdtypeuser, inar2);
            out.print(outdata2[2]);
            error=BEAN_TICU.GETERROR_MSSQL();
            if (error[0]!=null){
            out.print("Excepção do SQL Server: "+error[0]);
            } else if (error[1]!=null){
            out.print("Erro do SQL Server ");
            out.print("Numero de Erro: "+error[1]);
            out.print("Gravidade: "+error[2]);
            out.print("Estado: "+error[3]);
            out.print("Procedimento: "+error[4]);
            out.print("Linha: "+error[5]);
            out.print("Mensagem: "+error[6]);
            }
```

Figura 4.14– Invocação das funções de uma JavaBean

É dada depois ao utilizador a opção de ver dados de um determinado registo, alterar os seus dados ou ainda elimina-lo.

4.6.2. Ver Detalhes

Ao escolher está opção serão apresentados dados do registo escolhido.

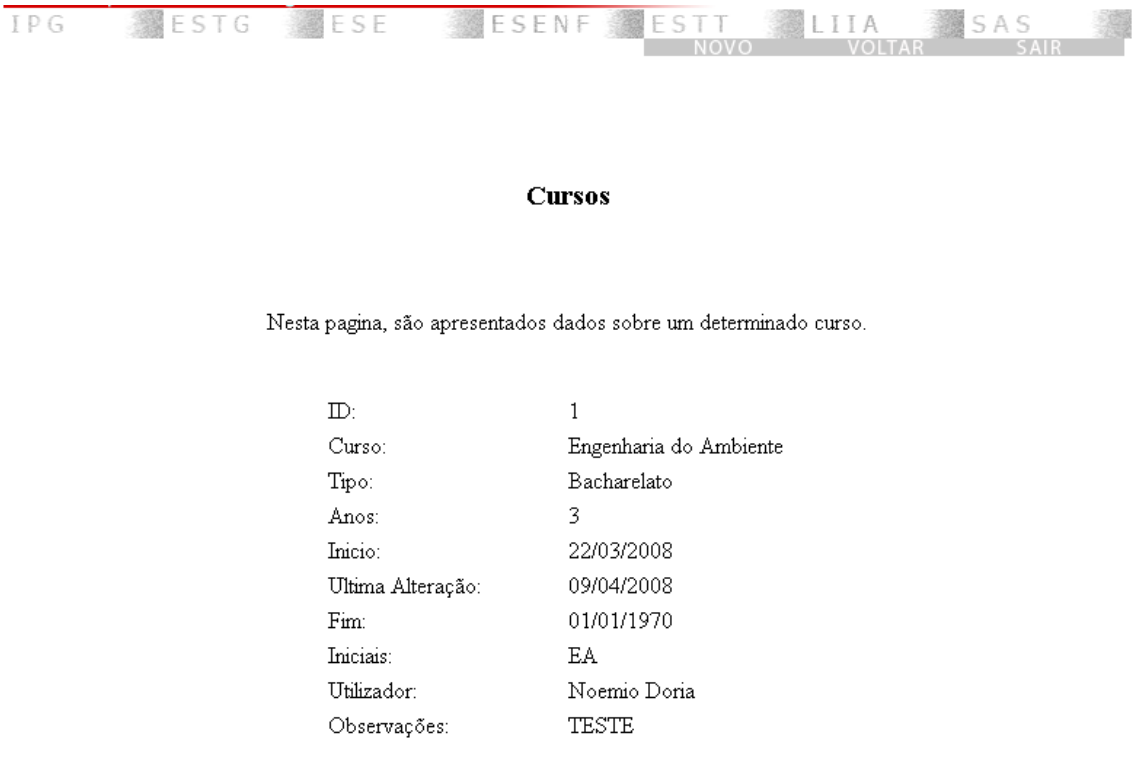

Figura 4.15– Obtenção de dados de um registo.

A pagina JSP faz novamente um pedido para a obtensão de dados do registo pedido

```
if (BD==1){
    inar[0]=iddis;
    outdata = BEAN_CURS.OBTCURS_MSSQL(bdtypeuser, inar);
    error=BEAN_CURS.GETERROR_MSSQL();
    if (error[0]!=null){
        out.print("Excepção do SQL Server: "+error[0]);
    } else if (error[1]!=null){
        out.print("Erro do SQL Server ");
        out.print("Numero de Erro: "+error[1]);
        out.print("Gravidade: "+error[2]);
        out.print("Estado: "+error[3]);
        out.print("Procedimento: "+error[4]);
        out.print("Linha: "+error[5]);
        out.print("Mensagem: "+error[6]);
    }
}
```

Figura 4.16– Código referente à obtenção de um registo.

Neste caso (Figura 4.16), são obtidos dados de um determinado curso.

Contudo, há atributos que são representados pelos numero de identificação (ID). Assim a pagina JSP pede os dados dos registos cujos ID's fazem parte do registo.

```
<tr>
    <td height="21" valign="top">Curso:</td>
    <td valign="top"><%out.print(outdata[5]);%></td>
</tr>
<tr>
    <td height="21" valign="top">Tipo:</td>
    <td valign="top"><%
    //out.print(outdata[4]);
          inar2[0] = outdata[4];
          if (BD==1){
          outdata2 = BEAN_TICU.OBTTICU_MSSQL(bdtypeuser, inar2);
          out.print(outdata2[2]);
          error=BEAN_TICU.GETERROR_MSSQL();
          if (error[0]!=null){
          out.print("Excepção do SQL Server: "+error[0]);
          } else if (error[1]!=null){
          out.print("Erro do SQL Server ");
          out.print("Numero de Erro: "+error[1]);
          out.print("Gravidade: "+error[2]);
          out.print("Estado: "+error[3]);
          out.print("Procedimento: "+error[4]);
          out.print("Linha: "+error[5]);
          out.print("Mensagem: "+error[6]);
          }
```

Figura 4.17– Código referente à obtenção de um registo adjacente.

Na Figura 4.17 representa-se o código para pedir dados sobre o tipo de curso.

Figura 4.17 – Código referente à obtenção de um registo adjacente.

```
<tr>
  <td height="21" valign="top">Utilizador:</td>
  <td valign="top">
      <%
      //out.print(outdata[3]);

          inar2[0] = outdata[3];

          if (BD==1){
          outdata2 = BEAN_UTIL.OBTUTIL_MSSQL(bdtypeuser, inar2);
          out.print(outdata2[5]+" "+outdata2[6]);
          error=BEAN_UTIL.GETERROR_MSSQL();
          if (error[0]!=null){
          out.print("Excepção do SQL Server: "+error[0]);
          } else if (error[1]!=null){
          out.print("Erro do SQL Server ");
          out.print("Numero de Erro: "+error[1]);
          out.print("Gravidade: "+error[2]);
          out.print("Estado: "+error[3]);
          out.print("Procedimento: "+error[4]);
          out.print("Linha: "+error[5]);
          out.print("Mensagem: "+error[6]);
          }

          }
```

Figura 4.18– Código referente à obtenção de um registo adjacente.

Neste caso, o Utilizador que procedeu à ultima alteração.

4.6.2. Editar e Novo

Ao optar por editar são mostrados os dados originais são mostrados e é mostrado o formulario para a alteração de dados. Avalidação dos dados é efectuada atraves de JavaScript.

Há atributos que são mostrados no formulario em forma de Combo-Box que são inseridos na mesma depois da pagina JSP ter requisitado o mesmos à camada central.

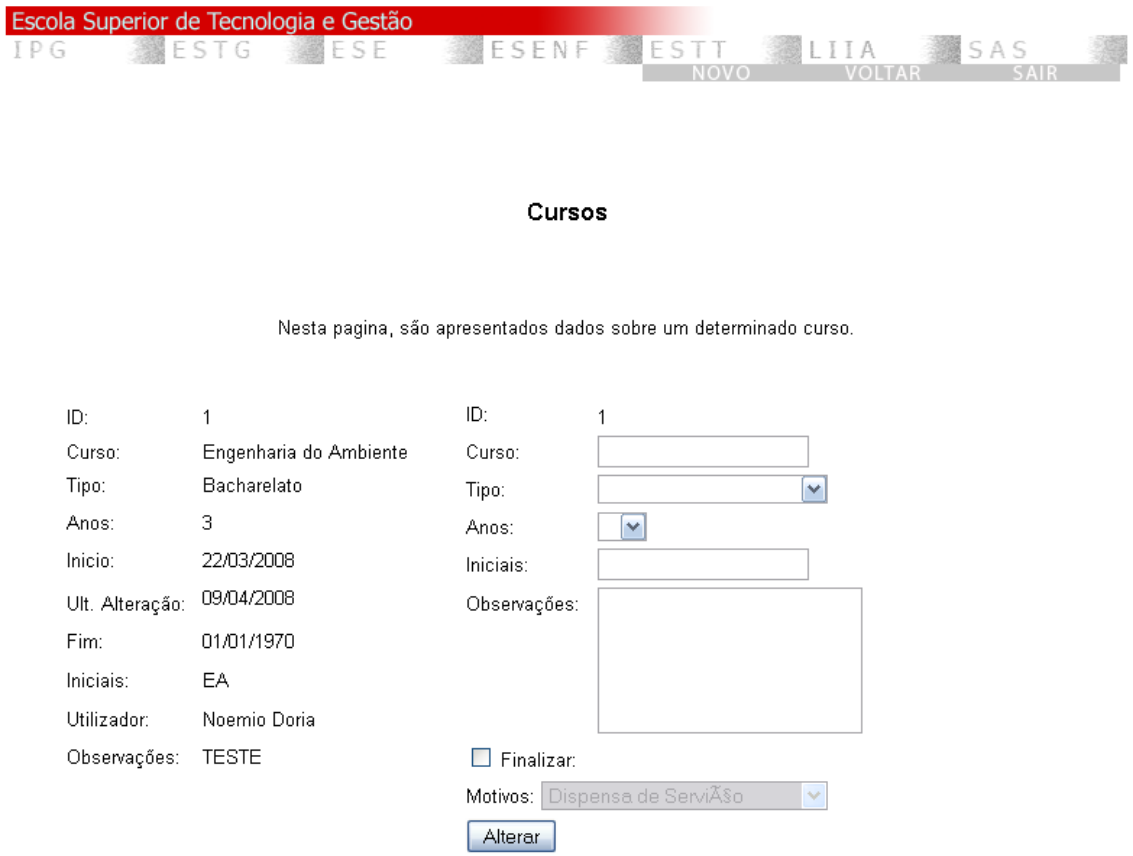

Figura 4.19– Formulário para editar dados de um curso.

O mesmo passa-se com a pagina JSP de inserção é apenas mostrado o formulário para a inserção de dados e as Combo-Boxes são preenchidos da mesma forma que a anterior.

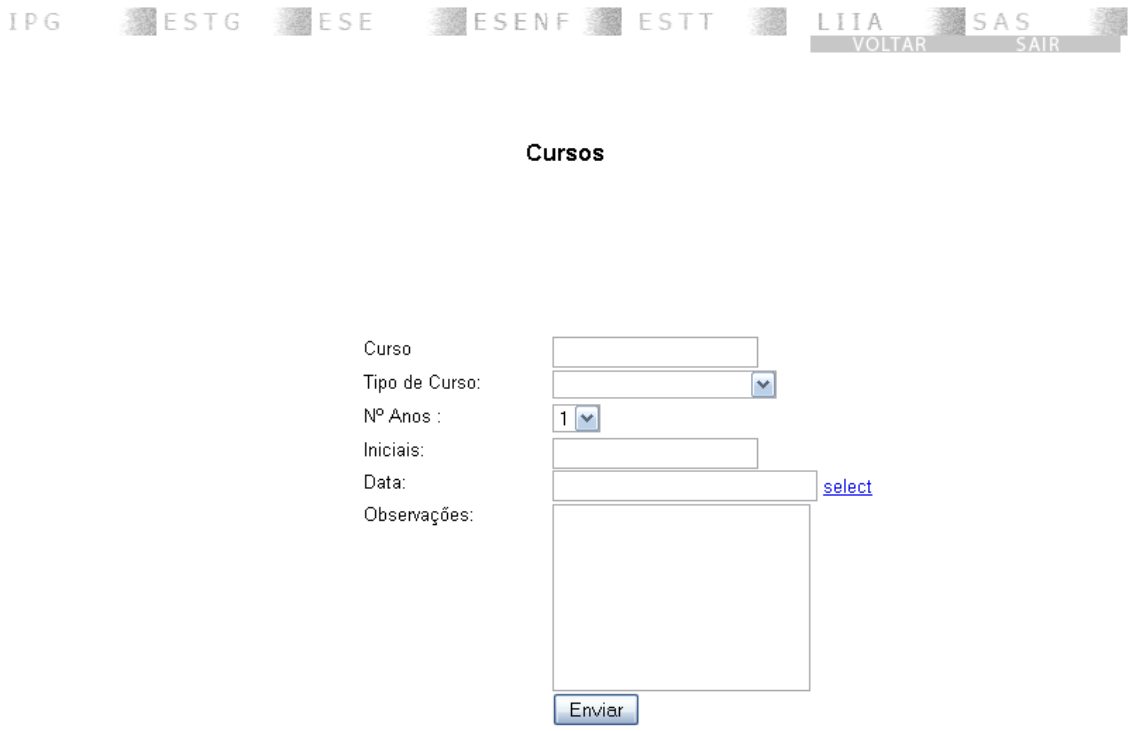

Figura 4.20– Formulário para inserir dados de um curso.

4.6.3. Validação

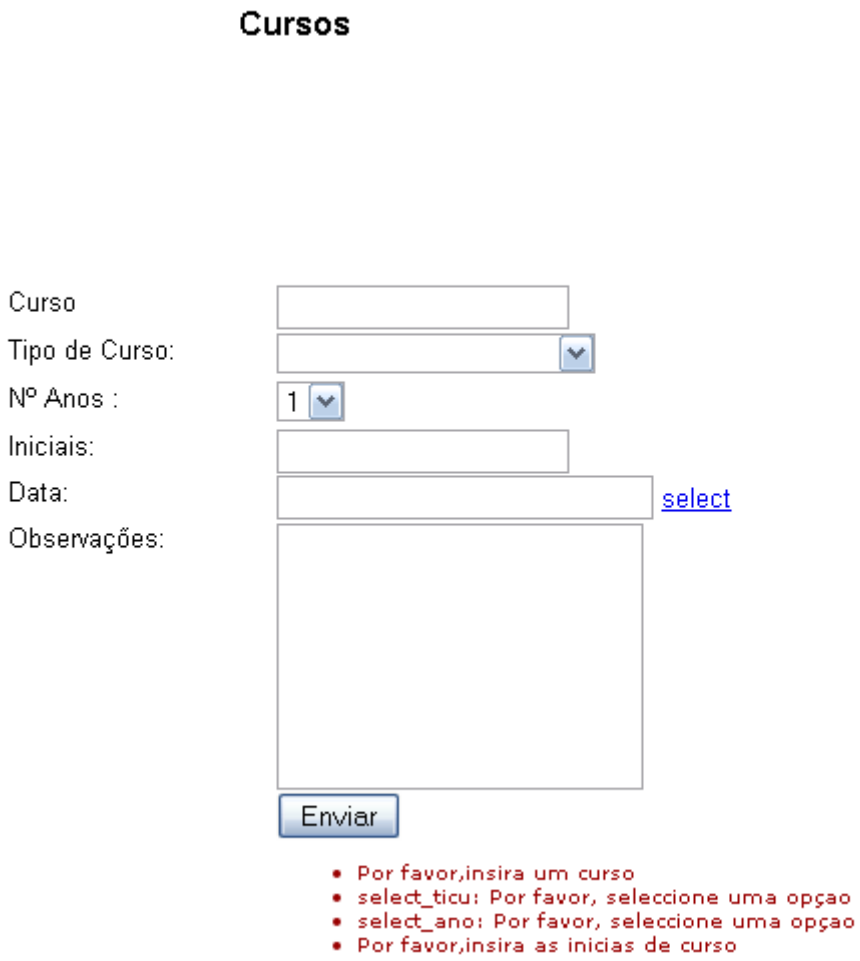

Figura 4.21– Formulário para editar dados de um curso com as validações activadas.

A validação como dito anteriormente a validação está a cargo de funções JavaScript que verificam o estado dos campos.

4.6.4. Inserção de Horários

Como o segundo objectivo foi a criação de horários decidiu-se aqui elucidar a implementação do processo de inserção de registos para a tabela Horários.

A implementação começou com um fluxograma apresentado de seguida. Sendo esta a tabela mais importante pode ver-se que existem diversas tabelas que lhe são adjacentes.

O primeiro fluxograma exemplifica como a aplicação envia os dados para a 3ª camada e o segundo como são tratados os dados dentro dessa camada

Mostra-se também a interface de inserção e resultado.

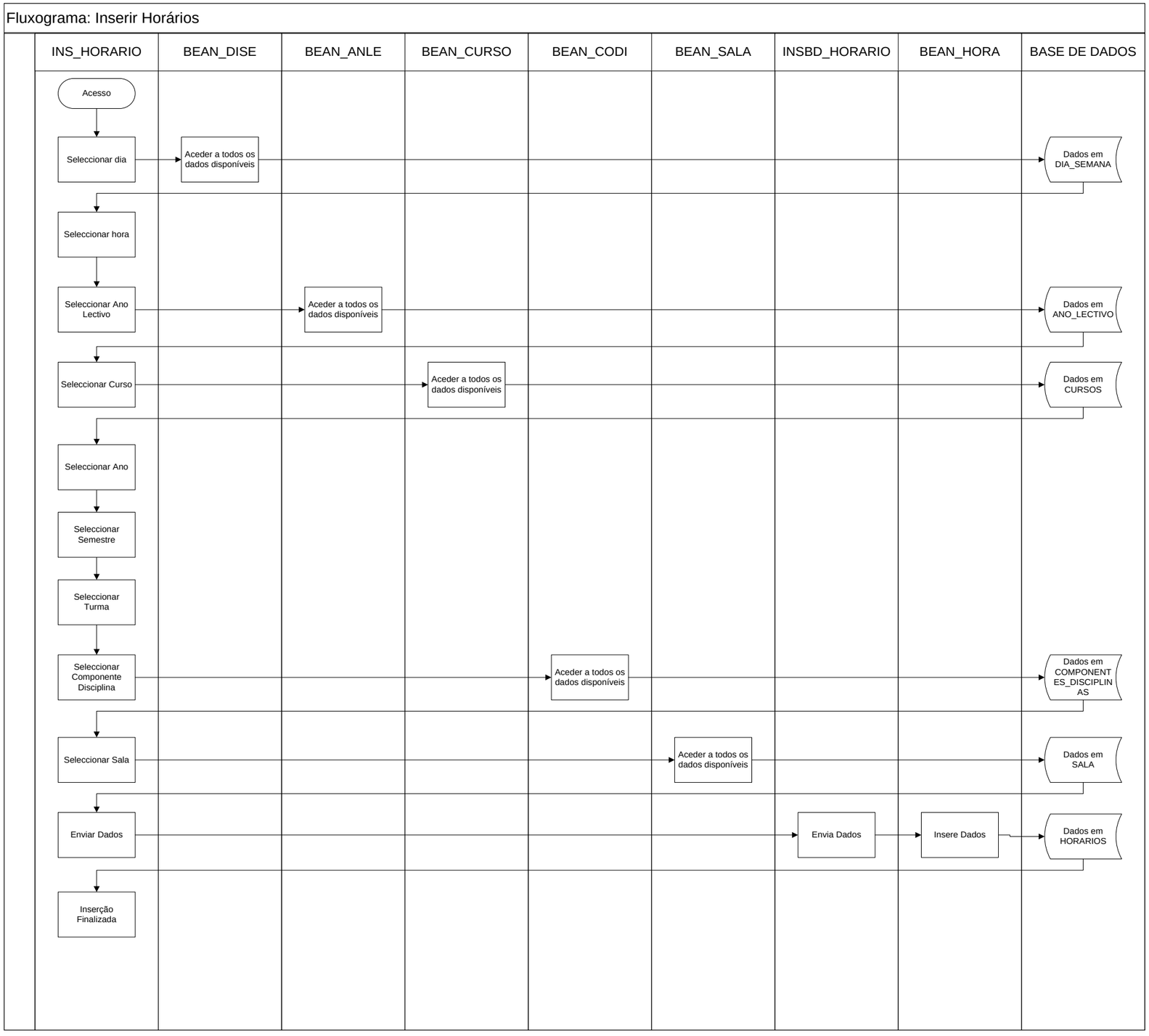

Figura 4.22– Fluxograma da aplicação para inserção de registos na tabela Horários.

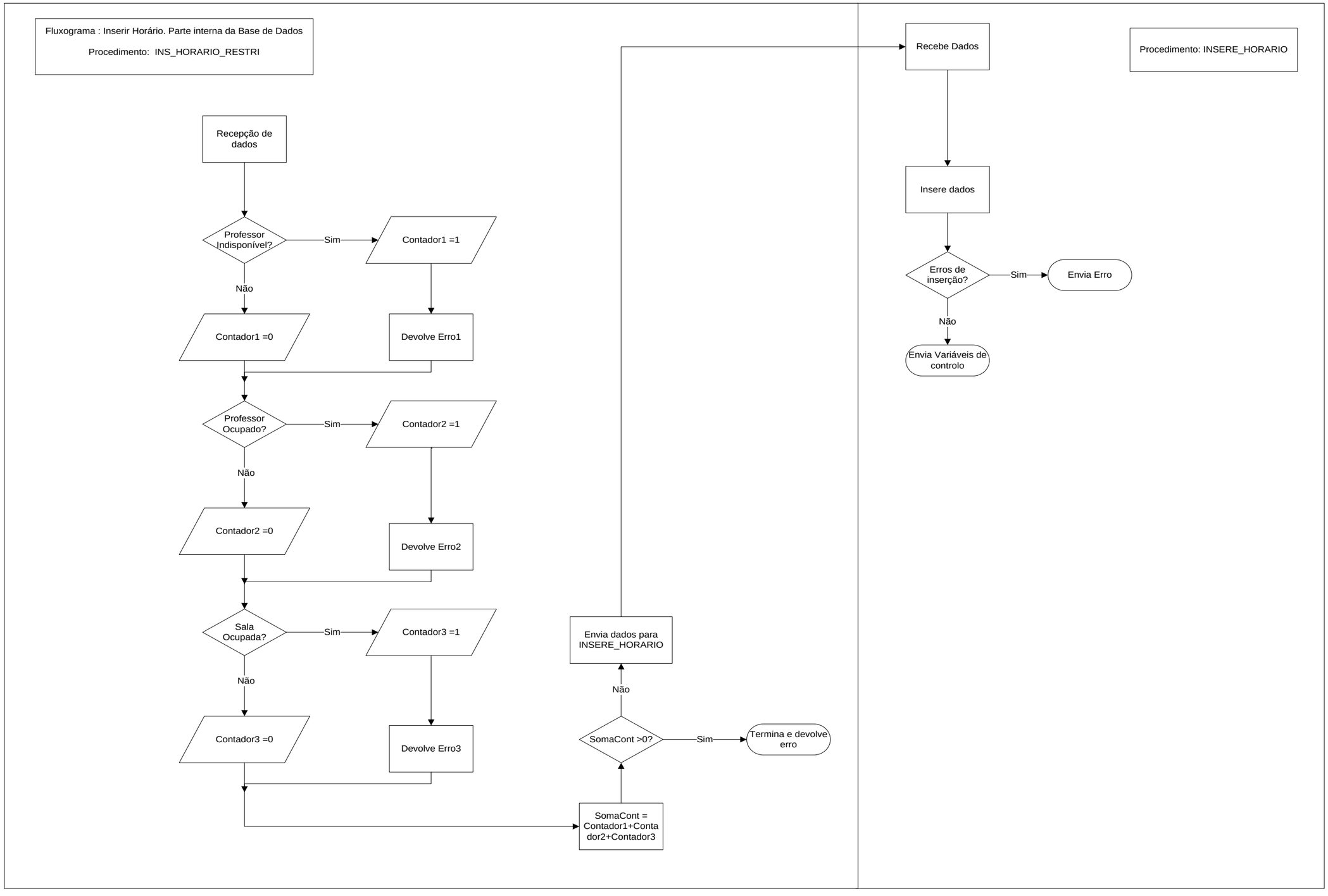

Figura 4.23– Fluxograma da base de dados para inserção de registos na tabela Horário

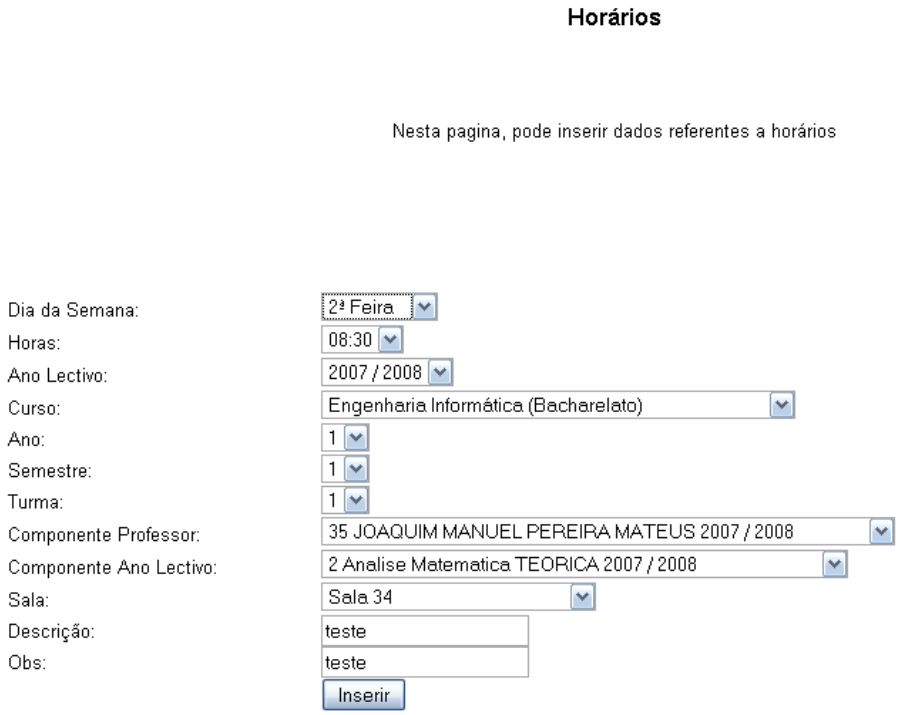

Figura 4.24– Interface para a inserção de dados

Figura 4.25. – Interface mostra um inserção de dados na tabela Horários bem sucedida.

4.7. Aspectos a considerar

Antes da instalação do Oracle 10gR2 é aconselhável a criação de uma placa virtual e proceder à sua configuração em computadores sem conexão à rede ou com conexão através de DHCP.

Assim sendo serão demonstrados o passos para tal processo.

Figura 4.26– Selecção de "Adicionar hardware".

Através de "Adicionar hardware", no painel de controlo, iniciou-se o processo.

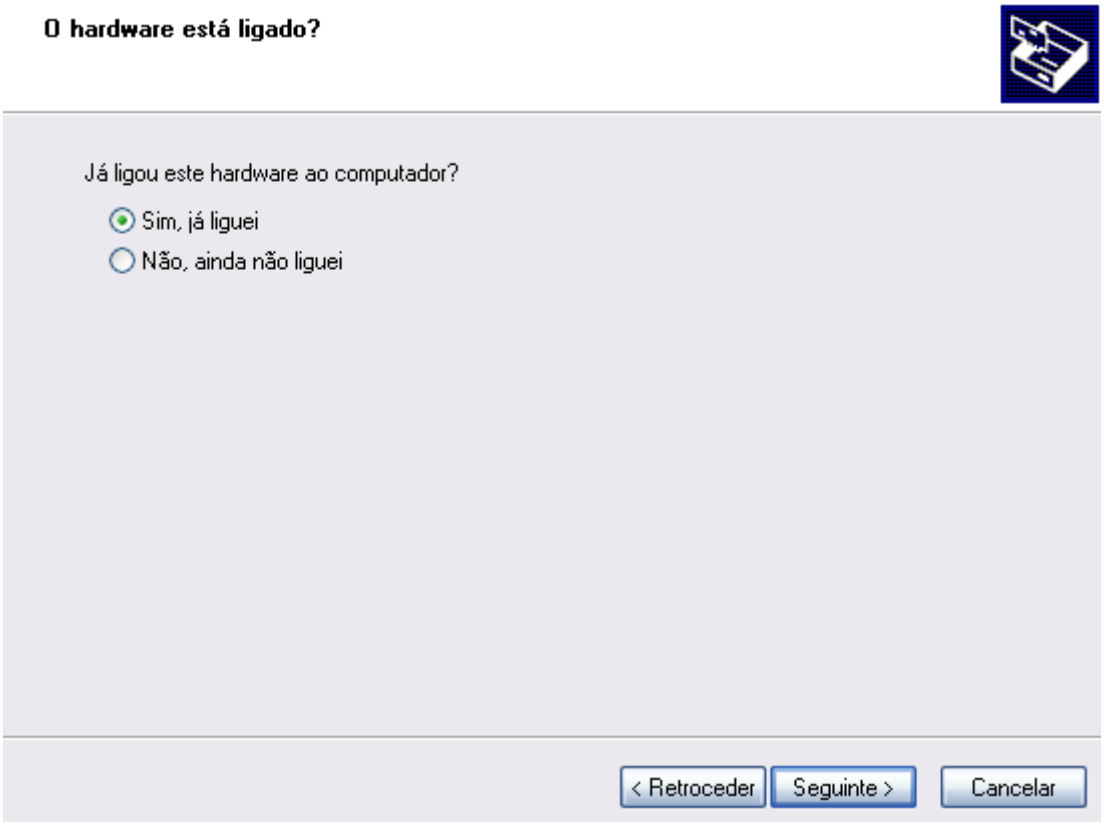

Figura 4.27– Indicação de hardware já conectado.

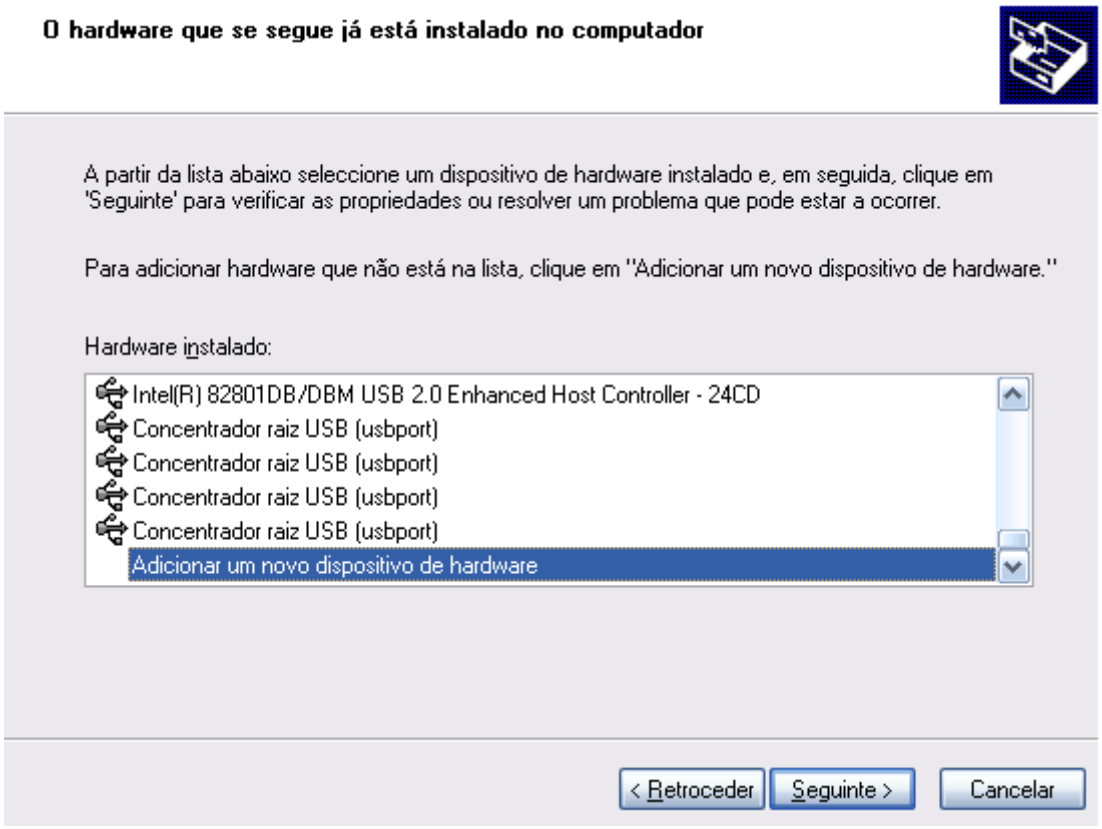

Figura 4.28. – Selecção do *driver* do hardware a adicionar.

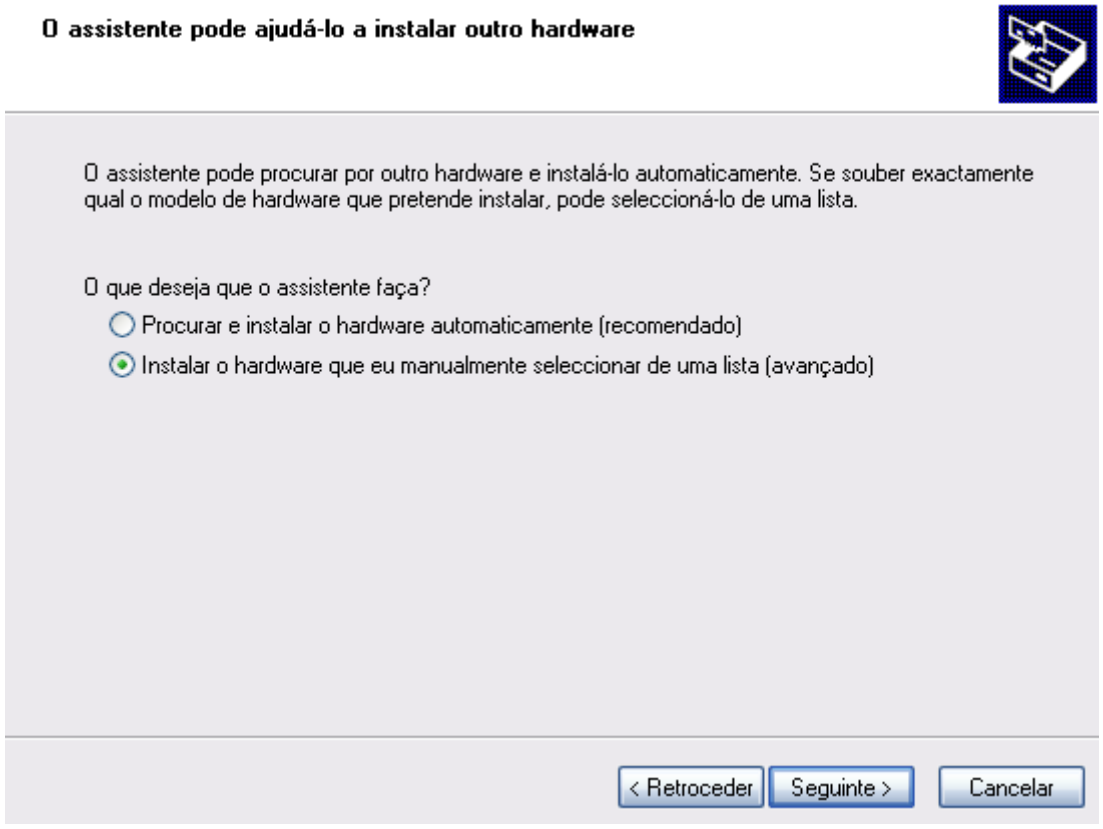

Figura 4.29. – Selecção de instalação manual de hardware.

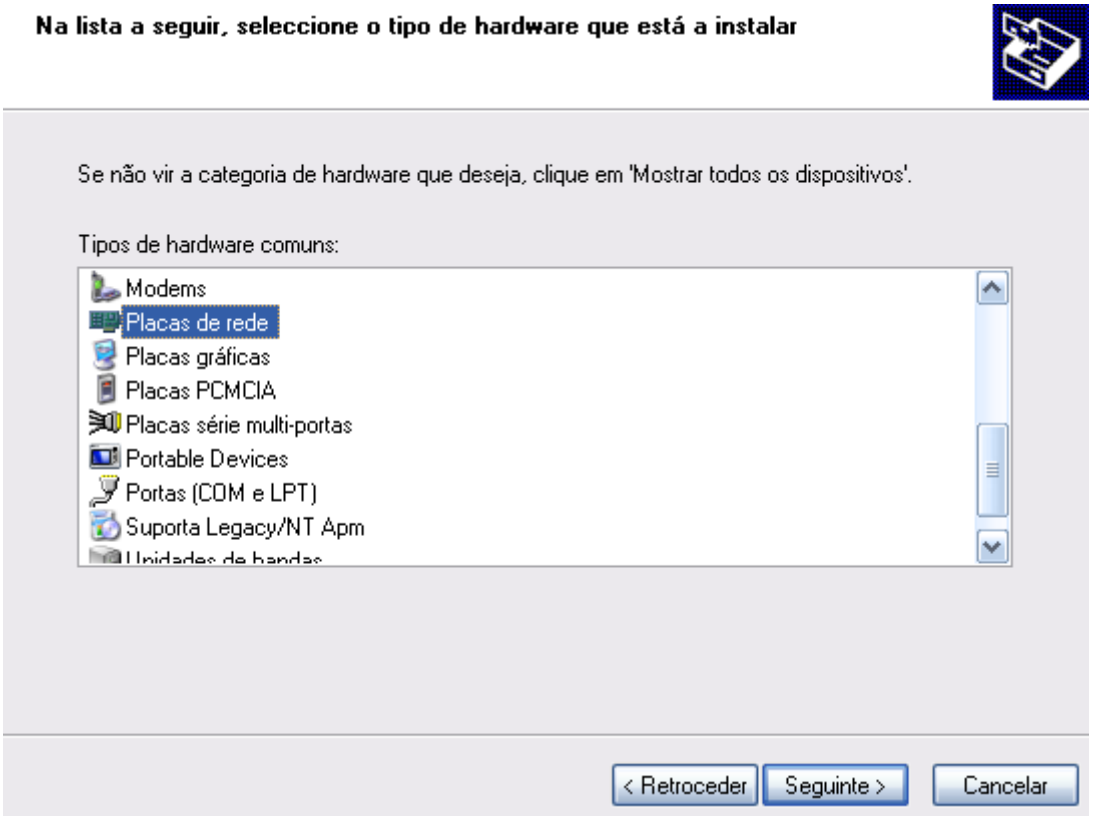

Figura 4.30– Selecção de "Adicionar hardware".

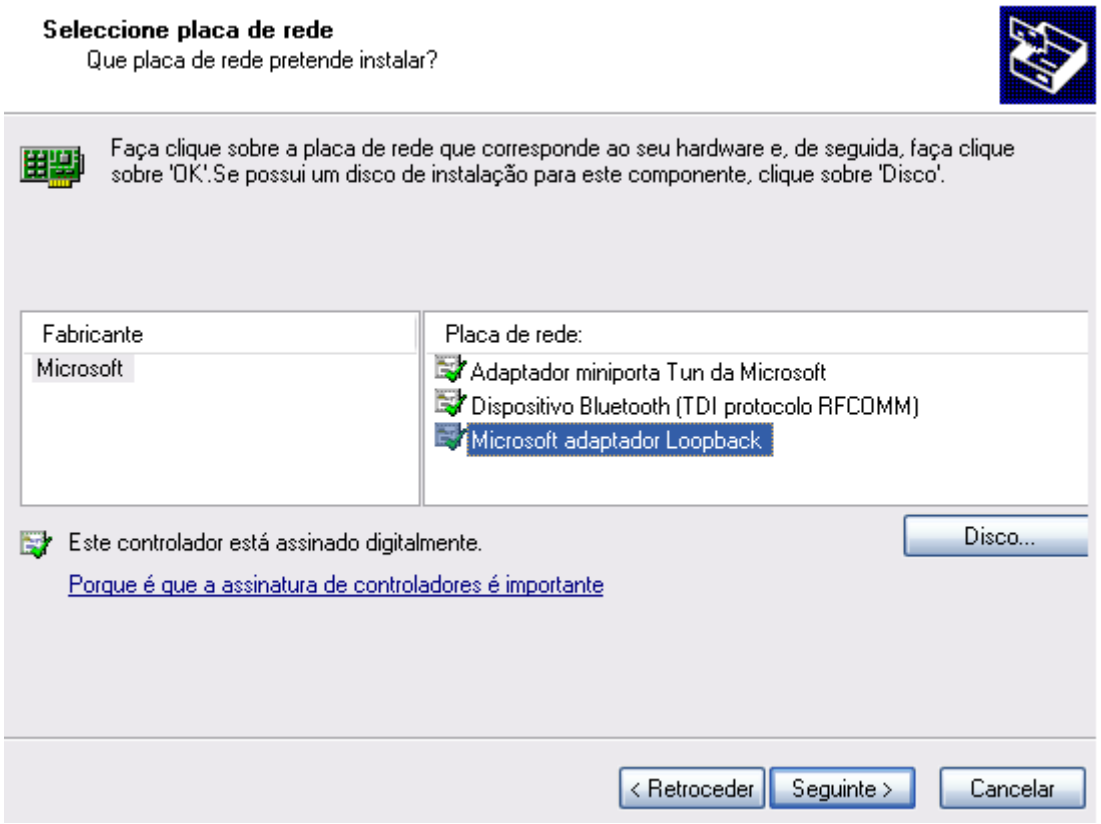

Figura 4.31– Selecção da placa de rede.

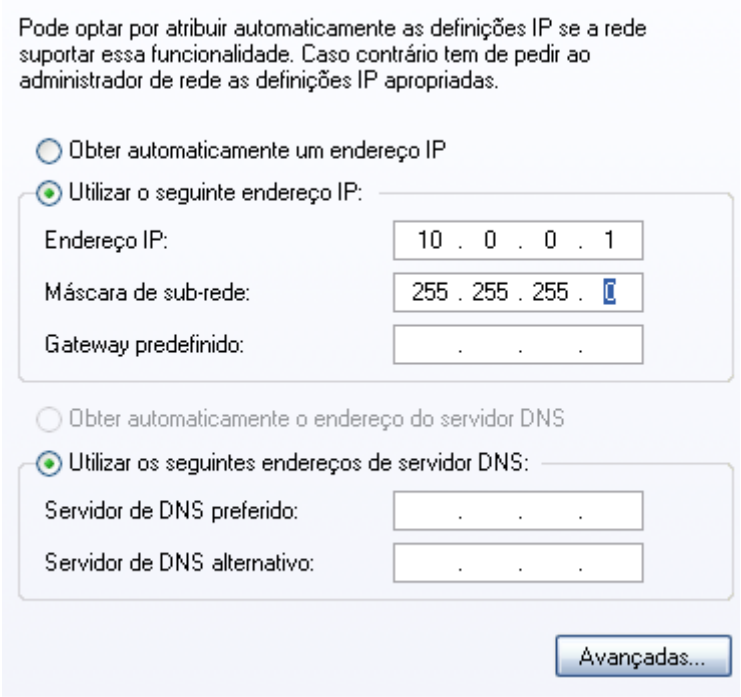

Figura 4.32. – Configuração da rede.

Para armazenar a informação referente às conexões às bases de dados foi criado um ficheiro XML que armazenou-se no disco rígido.

Figura 4.33– Ficheiro XML.

O acesso a esta informação fez-se através do *driver* JDOM na versão 1.0

Uma vez que foram utilizados *drivers jar* para a criação da aplicação aqui é mostrada a forma como foram associados à mesma.

Inicialmente acedeu-se às propriedades da aplicação e de seguida na secção das livrarias (*Iivraries*) inseriu-se o caminho para os mesmos.

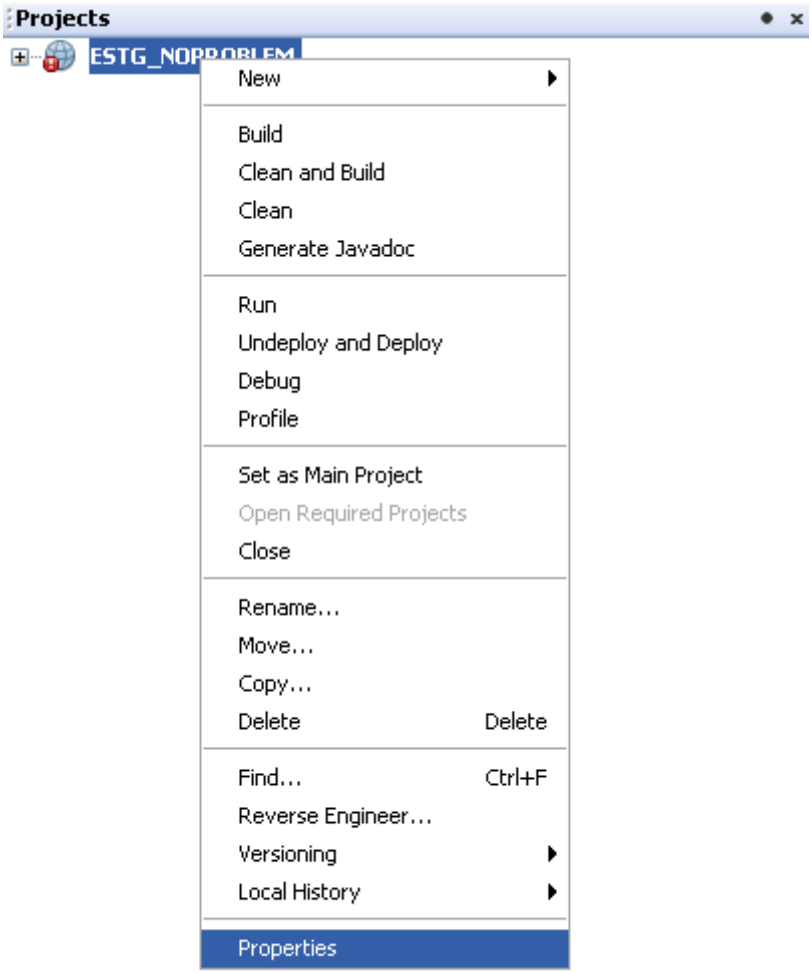

Figura 4.34– Acesso às propriedades.

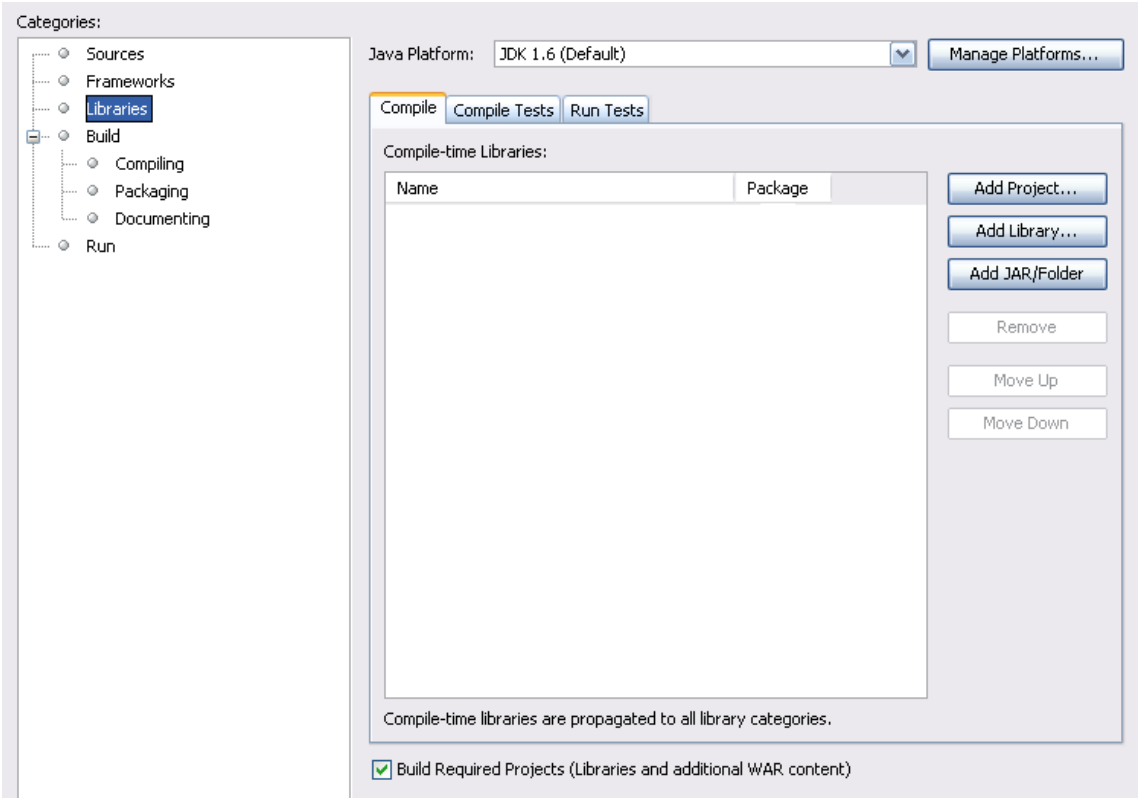

Figura 4.35– Acesso à secção de livrarias.

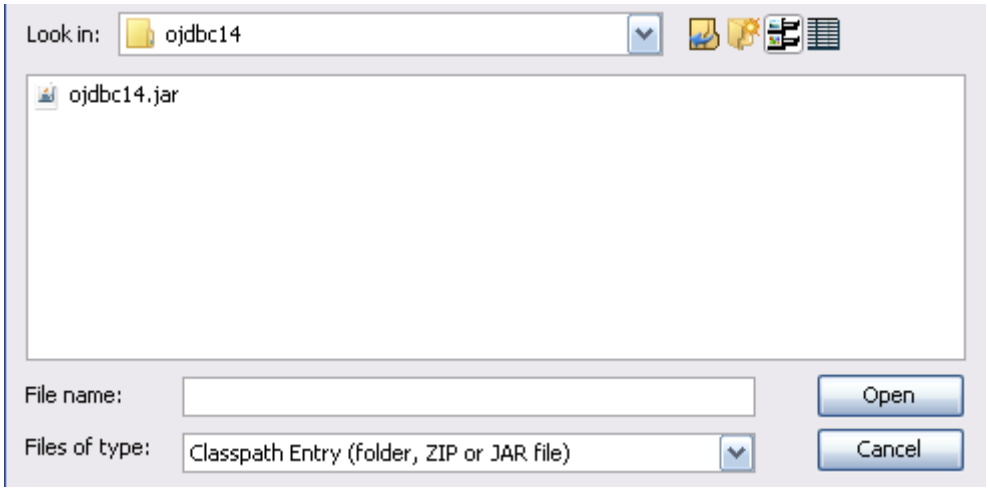

Figura 4.36– Selecção do *driver*.

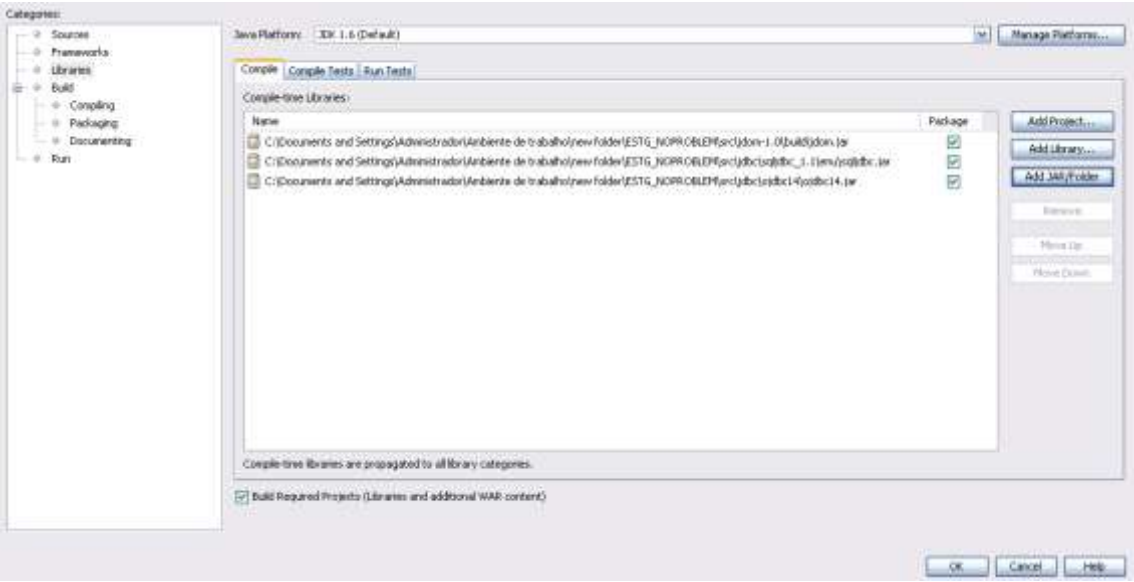

Figura 4.37– Selecção de *drivers* concluída

4. Testes e Análise de resultados

Aqui serão descritos alguns teste efectuados não se mostram todos uma vez que todos são de todo semelhantes com a excepção do interface dos horários.

Figura 4.1– Entrada no sistema.

O utilizador deve autenticar-se no sistema e pode usar qualquer uma das bases de dados.

Figura 4.2. – Interface mostra a entrada bem sucedida.

O sistema dá as boas-vindas ao utilizador e o mesmo deve clicar em "Continuar" para completar a sua entrada senão após 5 segundos será reencaminhado para a entrada.

Figura 4.3 – Interface mostra a entrada bem sucedida.

A validação de formulários é feita através de *JavaScript*.

Figura 4.4. – Interface mostra a entrada mal sucedida.

No caso de o utilizador não pertencer ao sistema informa-o e reencaminha-o para a entrada.

É mostrado o menu principal e o utilizador pode escolher para onde se deslocar. Neste caso será mostrada a obtenção de um horário.

Figura 4.5– Selecção de horário.

O utilizador escolhe o horário a visualizar.

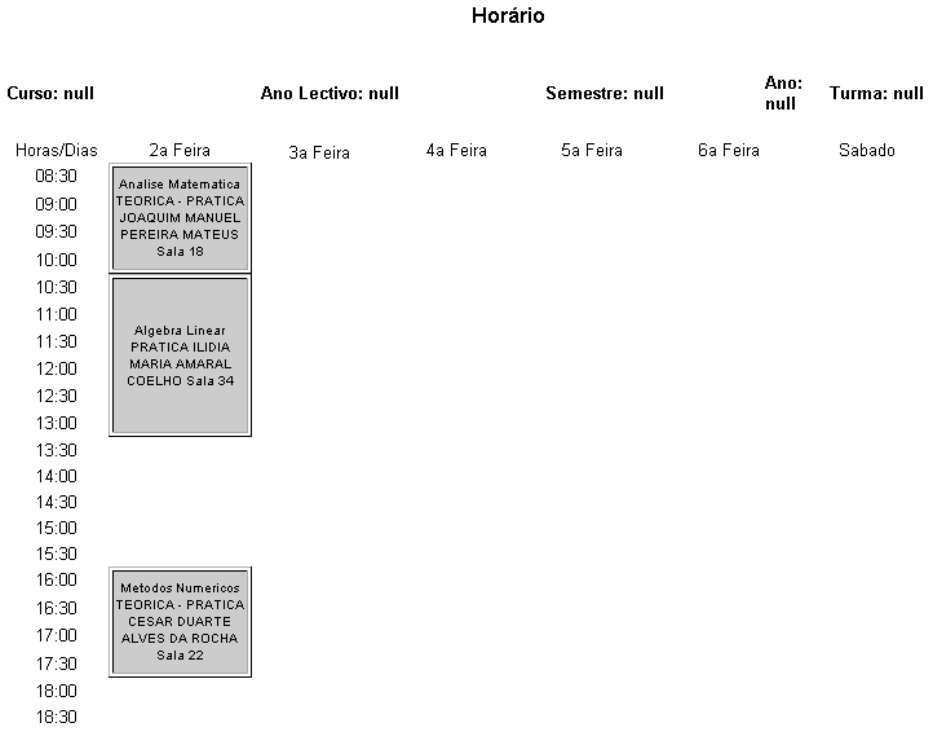

Figura 4.6– O interface mostra o horario

Neste caso, é apenas mostrado o horário da 2ª feira porque só existem registos desse dia e porque aplicação ainda está a sofrer testes.

5. Conclusão

A implementação do projecto descrito mostrou-se verdadeiramente desafiante. A análise de sistema mostrou-se como um processo longo e demorado que contudo, acabou por mostrar-se bastante abrangente.

Durante a implementação houveram certas classes que sofreram alterações de forma a integrarem-se melhor no sistema de forma a atingir os objectivos propostos.

De uma forma geral, conclui-se que a integração das bases de dados foi conseguida de uma forma clara e funcional. Embora inicialmente tivessem sido encontrados problemas no que toca às conexões JDBC o mesmo acontecendo algumas vezes que a aplicação fora transladada de formatação para formatação do PC onde se encontrava. Esta abordagem simplista poderia ser substituída por outras já existentes de forma a facilitar as interacções da aplicação com as bases de dados. Pode considerar-se assim o 1º objectivo alcançado.

Quanto a aplicação em si encontra-se ainda em testes e a sofrer algumas alterações quando este relatório estava a ser redigido. Por isso, pensa-se que estará à altura dos objectivos propostos permitindo uma utilização fácil como ferramenta de apoio para a criação de horários..

A vantagem desta aplicação passa pela portabilidade podendo ser acedida através qualquer *Web browser*. A desvantagem é que sem ligação à Internet não é possível aceder à aplicação.

Para desenvolvimento futuro, propõe a optimização das bases de dados para permitirem um acesso mais rápido. A reestruturação da aplicação de forma a melhorar o seu desempenho. O histórico poderá mais tarde ser armazenado num ficheiro. De forma a impedir a seu crescimento.

Deverá também desenvolver-se a opção de impressão através da aplicação que é uma opção na aplicação que ficaram em *stand by*

6. Bibliografia

cocoalab. *Model View Controller.* 10 de Julho de 2007. http://www.cocoalab.com/?q=node/24 (acedido em 29 de Novembro de 2008).

Nogueira, Admilson. *Histórico da UML.* 21 de Fevereiro de 2005. http://imasters.uol.com.br/artigo/2994/uml/historico_da_uml/ (acedido em 29 de Novembro de 29).

Wikipedia. *Java (linguagem de programação).* 11 de Dezembro de 2008. http://pt.wikipedia.org/wiki/Java_(linguagem_de_programa%C3%A7%C3%A3o) (acedido em 5 de Dezembro de 2008).

—. *JavaScript.* 17 de Dezembro de 2008. http://en.wikipedia.org/wiki/JavaScript#Structured_programming (acedido em 03 de Dezembro de 2008).

—. *JavaServer Pages.* 12 de Dezembro de 2008. http://en.wikipedia.org/wiki/Javaserver_pages (acedido em 29 de Novembro de 2008).

—. *Microsoft SQL Server.* 14 de Dezembro de 2008. http://pt.wikipedia.org/wiki/Sql_server (acedido em 29 de Novembro de 2008).

—. *MVC.* 11 de Dezembro de 2008. http://pt.wikipedia.org/wiki/MVC (acedido em 09 de Dezembro de 2008).

—. *Oracle.* 2 de Dezembro de 2008. http://pt.wikipedia.org/wiki/Oracle (acedido em 2 de Dezembro de 2008).

—. *SQL.* 12 de Dezembro de 2008. http://pt.wikipedia.org/wiki/Sql (acedido em 29 de Novembro de 2008).

—. *UML.* 9 de Dezembro de 2008. http://pt.wikipedia.org/wiki/Unified_Modeling_Language (acedido em 10 de Dezembro de 2008).