

**Trabalho Fase 1 – Projeto “GameOn”**

Vasco Branco – 48259

João Pereira – 48264

Tiago Neves – 48292

|  |  |
| --- | --- |
| Orientadores | Prof. Walter Vieira |

Relatório final realizado no âmbito de Sistemas de Informação,  
do curso de licenciatura em Engenharia Informática e de Computadores  
Semestre de Verão 2022/2023

Maio de 2013

# Resumo

Neste trabalho, o grupo recebeu um texto como enunciado e é desafiado a criar um modelo entidade-associação com base nesse texto. Esse modelo permite identificar as entidades presentes no texto e estabelecer as associações entre elas.

Em seguida, o grupo teve que utilizar o modelo entidade-associação criado para desenvolver um modelo relacional. Esse modelo relacional representa as entidades, seus atributos e os relacionamentos entre elas em um formato que pode ser implementado em um banco de dados relacional.

Além disso, o grupo precisa de utilizar a uma extensão para programação da linguagem SQL, denominada por PLPGSQL, para implementar diversas functions, procedures, views e triggers. Esses elementos adicionam funcionalidades ao banco de dados e permitem realizar operações complexas e automatizadas.

Durante o desenvolvimento desses componentes, o grupo aplicou os corretos níveis de isolamento, garantindo a integridade dos dados e evitando conflitos de acesso simultâneo. Além disso, foram aplicados os devidos tratamentos de erros para lidar com situações inesperadas e manter a consistência dos dados.

Em resumo, este trabalho envolve a criação de um modelo entidade-associação a partir de um texto, a tradução desse modelo para um modelo relacional, e a implementação de functions, procedures, views e triggers utilizando a extensão para programação da linguagem sql, com atenção aos níveis de isolamento e tratamento de erros.

**Palavras-chave: Extensão** PLPGSQL**; Function; Modelo entidade-associação; Modelo relacional; Níveis de isolamento; Procedures; Tratamento de erros; Triggers; Views**

# Abstract

Abstract text (1 page).

**Keywords:** sorted keyword list, delimited by ;.

**Índice**

[Resumo v](#_Toc417484088)

[Abstract vii](#_Toc417484089)

[Agradecimentos ix](#_Toc417484090)

[Lista de Figuras xiii](#_Toc417484091)

[Lista de Tabelas xv](#_Toc417484092)

[1. Introdução 1](#_Toc417484093)

[1.1 Nome da secção deste capítulo 1](#_Toc417484094)

[1.2 A segunda secção deste capítulo 1](#_Toc417484095)

[1.2.1 A primeira sub-secção desta secção 1](#_Toc417484096)

[1.2.2 A segunda sub-secção desta secção 1](#_Toc417484097)

[1.3 Organização do documento 2](#_Toc417484098)

[2. Formulação do Problema 3](#_Toc417484099)

[2.1 Nome da secção deste capítulo 3](#_Toc417484100)

[2.2 Análise do problema - enumeração 3](#_Toc417484101)

[2.3 Outro problema - tabela 4](#_Toc417484102)

[2.4 Expressões matemáticas 4](#_Toc417484103)

[2.5 Figuras de grande dimensão 4](#_Toc417484104)

[3. Solução Proposta - Grandes Ideias 7](#_Toc417484105)

[3.1 Nome da primeira secção deste capítulo 7](#_Toc417484106)

[3.2 A segunda secção deste capítulo 8](#_Toc417484107)

[3.2.1 A primeira sub-secção desta secção 8](#_Toc417484108)

[3.2.2 A segunda sub-secção desta secção 8](#_Toc417484109)

[3.3 Descrição detalhada da solução 8](#_Toc417484110)

[4. Avaliação Experimental 11](#_Toc417484111)

[4.1 Nome da primeira secção deste capítulo 11](#_Toc417484112)

[4.2 A segunda secção deste capítulo 11](#_Toc417484113)

[4.2.1 A primeira sub-secção desta secção 11](#_Toc417484114)

[4.2.2 A segunda sub-secção desta secção 11](#_Toc417484115)

[4.3 Análise de resultados 11](#_Toc417484116)

[5. Conclusões 13](#_Toc417484117)

[Referências 14](#_Toc417484118)

[A.1 Diagramas da Aplicação 15](#_Toc417484119)

[A.2 Modelos de dados 17](#_Toc417484120)

# 

# Lista de Figuras

[Figura 1 - Legenda da figura com o logotipo do ISEL. 1](#_Toc416101905)

[Figura 2 - Legenda da figura com o logotipo do ISEL – versão 2. 5](#_Toc416101906)

[Figura 3 – Diagrama de casos de utilização. 15](#_Toc416101907)

# Lista de Tabelas

[Tabela 1 -Um exemplo de legenda de tabela. Prazos de entrega de Projecto e Seminário, 4](#_Toc416101908)

# Introdução

Capítulo 1: Modelos

Introdução: O Capítulo 1 do relatório aborda a etapa de criação do banco de dados, que é essencial no processo de desenvolvimento de sistemas. Neste capítulo, serão apresentados os dois subcapítulos que abordam os principais modelos utilizados nesse processo: o Modelo Entidade-Associação e o Modelo Relacional. Esses modelos fornecem uma representação estruturada das entidades, atributos e relacionamentos presentes no sistema, permitindo uma melhor compreensão e organização dos dados.

## 1.1 Modelo Entidade-Associação

O modelo entidade-associação é uma ferramenta utilizada para representar as entidades presentes em um sistema, bem como os relacionamentos entre elas. Neste subcapítulo, será abordada a criação do modelo entidade-associação com base no texto fornecido como enunciado do trabalho.

O processo de criação do modelo entidade-associação envolve a identificação das entidades presentes no texto e a definição dos relacionamentos entre elas. As entidades representam objetos ou conceitos do mundo real, enquanto os relacionamentos representam as associações entre essas entidades.

Após identificar as entidades, é importante estabelecer os relacionamentos entre elas. Os relacionamentos podem possuir diferentes cardinais, como "um para um", "um para muitos" ou "muitos para muitos" ou até conterem obrigatoriedade, sendo esta representada com duas linhas paralelas entre si. Esses relacionamentos são representados por meio de linhas que conectam as entidades no diagrama.

Além disso, é necessário definir os atributos das entidades, que são características ou propriedades que descrevem cada entidade.

O modelo entidade-associação fornece uma representação visual das entidades, seus atributos e os relacionamentos entre elas, auxiliando na compreensão e na estruturação do banco de dados.

## 1.2 Modelo Relacional

o Modelo Relacional, que é uma representação estruturada das entidades, atributos e relacionamentos do sistema em formato de tabelas. Será explorado o processo de tradução do Modelo Entidade-Associação para o Modelo Relacional, onde as entidades se tornam tabelas, os atributos se tornam colunas e os relacionamentos são representados por meio de chaves primárias e estrangeiras.

Assim, o modelo relacional foi construído com base nas seguintes regras:

Passo 1: Entidade E 1. Entidade E → Relação T 2. Attrs(E) simples → Attrs(T) 3. Attrs(E) simples de 1 atributo composto → Attrs(T) 4. Atributo-chave de E → PK(T)

Resumo: Criar as tabelas que não são fracas juntamente com os respetivos atributos.

• Passo 2: Entidade fraca W com entidade identificadora E 1. T relação que representa E 2. Entidade W → Relação U 3. Atributos simples/compostos de W → Attrs(U) (2 e 3 de Passo 1) 4. PK(T) → FK(U) 5. PK(T) + Atributos da chave parcial de W → PK(U)

Resumo: Criar as tabelas fracas, juntamente com os seus atributos, colocando como foreign key e primary key a primary key da entidade não fraca.

• Passo 3A: (chave estrangeira) Associação-tipo 1:1 de R(E1,E2) (Mais utilizada, com exceção de condições especiais, reduz o no. de tabelas) 1. T1 e T2 são as relações que representam E1 e E2 2. Escolhemos uma das relações, e.g. T2 3. Se uma das entidades (e.g. E2) tem participação total em R, por regra essa é a relação (e.g. T2) 4. Atributos simples/compostos de R → Attrs(E2) (2 e 3 de Passo 1) 5. PK(T1) → FK(T2)

Resumo: Escolher uma associação com cardinalidade 1:1, escolher qualquer uma das entidades participantes e consecutivamente criar essa mesma tabela que foi escolhida juntamente com os seus atributos, colocando como foreign key a primary key da entidade que não foi escolhida, caso a relação tenha algum atributo esse mesmo atributo também deve ser incluído na criação da entidade escolhida.

• Passo 4A: (chave estrangeira) Associação-tipo 1:N de R(E1,E2) (Mais utilizada, com exceção de condições especiais, reduz o no. de tabelas) 1. T1 e T2 são as relações que representam E1 e E2 2. E1 tem cardinalidade N na associação → Relação T1 3. Atributos simples/compostos de R → Attrs(T1) (2 e 3 de Passo 1) 4. PK(T2) → FK(T1), ex: cada instância de E2 está associada no máximo a 1 instância de E1

Resumo: Escolher uma associação com cardinalidade 1:N, escolher a entidade que possui o cardinal N nessa mesma relação, criar essa mesma tabela que foi escolhida juntamente com os seus atributos, colocando como foreign key a primary key da entidade com o cardinal 1.

• Passo 5: Associação-tipo N:N de R(E1,E2) 1. T1 e T2 são as relações que representam E1 e E2 2. R → Relação S 3. Atributos simples/compostos de R → Attrs(S) (2 e 3 de Passo 1) 4. PK(T1) e PK(T2) → FK(S) 5. PK(T1) + PK(T2) → PK(S)

Resumo: Escolher uma associação com cardinalidade N:N, criar uma tabela para a relação em si, juntamente com os seus atributos, colocando como primary key e como foreign key as primary keys das entidades envolvidas nessa mesma relação.

# Formulação do Problema

Neste capítulo explica-se o problema/desafio proposto para a primeira fase do projeto da disciplina de Sistemas de Informação.

Como primeira fase do projeto, foi no pedido que construíssemos um sistema para a gestão de jogos, utilizadores e partidas de uma empresa: “*A empresa “GameOn” pretende desenvolver um sistema para gerir jogos, jogadores e as partidas que estes efetuam.*”.

## Análise do problema

Para atingir tal objetivo tivemos de realizar diversos ficheiros (scripts) com a utilização de várias ferramentas (views, functions, procedures, etc.), ficheiros esses que serão o alvo de avaliação.

Também realizámos os devidos modelos de dados, modelo EA e modelo ER anteriormente explicados, de forma a auxiliar na organização da solução por nós apresentada.

# Solução Proposta - Scripts

Nesta secção, é apresentado todos os mecanismos e processos que o grupo teve de efetuar para solucionar os problemas dispostos no enunciado, desde da alínea a) até à alínea o).

Estas soluções foram construídas à base de instruções “SQL” e da extensão para programar em sql denominada por “plpgsql”.

Sendo mais específico, todas as soluções da modelagem do modelo físico consistem em instruções de criação, remoção e alteração de tabelas. Já a lógica de implementação dos mecanismos é baseada em procedures, functions, view, e triggers.

Deste modo, foi possível automatizar todas as soluções dispostas no trabalho efetuado pelo grupo.

## 3.1 Conjunto de Scripts

**( a ) Criar o modelo físico (1 script autónomo):**

Através da instrução “create table” é possível gerar uma tabela para uma base de dados relacional.

A sintaxe simples desta instrução é: create table table\_name (parameter\_name parameter\_type);

É também importante de realçar que nesta instrução é possível definir as primary e foreign keys da mesma tabela , constraints e colocar flags nos valores, como por exemplo “not null”. Com esta instrução foi possível criar o modelo físico num único script autónomo.

**( b ) Remover o modelo físico (1 script autónomo):**

Através da instrução “drop” é possível remover uma tabela da base de dados relacional.

A sintaxe simples desta instrução é: drop table\_name;

É também importante de realçar que nesta instrução é possível colocar “cascade” que permite remover não só a tabela em questão como também mas também todos os objetos dependentes dela, como restrições de chave estrangeira, visualizações ou triggers. Outra funcionalidade utilizada foi a cláusula “if exists”, caso a tabela exista o comando é executado. Com esta instrução foi possível remover o modelo físico num único script autónomo.

**( c ) Preenchimento inicial da base de dados (1 script autónomo):**

Através da instrução “insert into” é possível gerar uma tabela para uma base de dados relacional.

A sintaxe simples desta instrução é: insert into table\_name values(parameter\_name).

Com esta instrução foi possível "povoar“ o modelo físico num único script autónomo.

**( d ) Criar os mecanismos que permitam criar o jogador, dados os seus email, região e username, desativar e banir o jogador:**

Para criar um mecanismo que implementa-se a funcionalidade de criar um jogador, recorreu-se à criação de um procedure, através do comando “create or replace procedure criar\_jogador(...)”, passando como parâmetro o email, username e nome\_regiao, sendo todos os dados do tipo varchar.

Dentro deste procedimento, verifica-se se o email ou o username já existem na tabela “JOGADORES” se esta condição for verdadeira então é executado a instrução “raise exception”. Caso Contrário o jogador é inserido na tabela com os respetivos dados passados como parâmetro através do comando “insert into(...) values(...)”.

Já para desativar e banir o jogador foi também criado procedures mas que só recebem um único parâmetro sendo este o id do jogador, sendo este um int. Ambos também realizam uma verificação, através doa instrução “if exists” para ver se o jogador existe, se o jogador não existir é executado o comando “raise exception”, caso contrário é feito um update na coluna “estado\_player” no registo que corresponder ao id do jogador, isto na tabela de “JOGADORES”.

No caso de desativar o jogador é colocado o valor ‘Inativo’ e no caso de banir o jogador é colocado o valor ‘Banido’.

**( e ) Criar, sem usar as tabelas de estatísticas, a função totalPontosJogador que recebe como parâmetro o identificador de um jogador e devolve o número total de pontos obtidos pelo jogador:**

Para criar uma função que implementa a funcionalidade de devolver o número total de pontos obtidos pelo jogador, recorreu-se à criação de uma function, através do comando “create or replace function totalPontosJogador(...), que recebe como parâmetro o id do jogador, sendo este um int. Dentro desta function, verifica-se se o id passado como parâmetro existe dentro na tabela de “JOGADORES” se não existir é executado o comando “raise exception”, caso contrário é efetuada uma leitura da tabela “NORMAL” , para esse mesmo id de jogador passado como parâmetro, que é “agregada” com o comando “union all” à leitura da tabela “JOGA\_MJ”, sendo ainda feita uma leitura posterior que conta o número de id’s distintos presentes na tabela e soma o número de pontos total desse mesmo jogador.

Depois é também efetuada uma verificação para confirmar se o id do player passado como parâmetro não existe na tabela de “ESTATISTICAS\_JOGADORES”, se a condição for verdadeira, então essa mesma tabela é povoada com os pontos do jogador.

A função retorna os pontos do jogador passado como parâmetro.

**( f ) Criar, sem usar as tabelas de estatísticas, a função totalJogosJogador que recebe como**

**parâmetro o identificador de um jogador e devolve o número total de jogos diferentes nos**

**quais o jogador participou:**

Para realizar esta alínea recorremos à criação de uma função, onde realizamos a associação de todos os jogos jogados por um jogador por meio do identificador desse mesmo jogador, retornando a contagem do total de jogos obtidos.

Realizamos também a verificação do identificador de jogador recebido como parâmetro da função, assegurando que o jogador existe na tabela “JOGADORES”.

**( g ) Criar a função PontosJogoPorJogador que recebe como parâmetro a referência de um jogo e devolve uma tabela com duas colunas (identificador de jogador, total de pontos) em que cada linha contém o identificador de um jogador e o total de pontos que esse jogador teve nesse jogo. Apenas devem ser devolvidos os jogadores que tenham jogado o jogo:**

Em primeiro lugar, na função realizada, verificamos se a referência de jogo passada como parâmetro existe na tabela “JOGOS”. Em seguida, procedemos à criação de uma tabela temporária e para preencher essa tabela realizamos uma união entre todos os jogadores presentes nas tabelas “NORMAL” e “JOGA\_MJ” que tenham a referência de jogo associada a eles, inserindo nessa tabela o seu identificador de jogador e a soma de pontos de ambas as tabelas, ou seja, o total de pontos que obteve no jogo, tanto em jogo normal (single-player) como em jogo multijogador (multi-player).

Nesta função também realizamos o preenchimento da tabela “ESTATISTICAS\_JOGO”.

**( h ) Criar o procedimento armazenado associarCrachá que recebe como parâmetros o**

**identificador de um jogador, a referência de um jogo e o nome de um crachá desse jogo e**

**atribui o crachá a esse jogador se ele reunir as condições para o obter:**

Nesta função, primeiro verificamos se todos os parâmetros passados existem nas suas respetivas tabelas, em seguida criamos uma tabela temporária onde vamos armazenar o total de pontos que o jogador do respetivo identificador de jogador passado nos parâmetros fez no respetivo jogo também passado nos parâmetros por via da sua referência. Finalmente, se o jogador atingir a condição definida por nós então o mesmo será associado ao crachá com o nome passado nos parâmetros, na tabela “TEM”.

**( i ) Criar o procedimento armazenado iniciarConversa iniciar uma conversa (chat) dados o identificador de um jogador e o nome da conversa. O jogador deve ficar automaticamente  
associado à conversa. O procedimento deve devolver num parâmetro de saída o identificador da conversa criada:**

Após a verificação dos parâmetros passados, procedemos para a criação da nova conversa, inserindo o nome passado na tabela “CONVERSAS”. Apenas precisamos de inserir o nome da conversa pois o seu identificador é gerado automaticamente nesta inserção.

Em seguida obtemos esse identificador e então associamos o jogador à conversa, inserido os dados do jogador e o identificador da conversa na tabela “CRIAR”.

Finalmente retornamos o identificador da conversa.

**( j ) Criar o procedimento armazenado juntarConversa que recebe como parâmetros os**

**identificadores de um jogador e de uma conversa e junta esse jogador a essa conversa:**

Sucintamente, neste procedimento obtemos os dados necessários para realizar a associação de um jogador a uma conversa por meio do seu identificador, passado como parâmetro, e depois inserimos esses mesmos dados juntamente com o identificador da conversa, também passado como parâmetro, na tabela “CRIAR”.

## 3.2 Níveis de isolamento:

Em SQL, os níveis de isolamento são mecanismos que controlam como as transações interagem entre si em um banco de dados relacional. Eles garantem a consistência e a integridade dos dados durante a execução de transações simultâneas.

Existem vários níveis de isolamento que podem ser configurados num banco de dados, sendo os mais comuns:

Read Uncommitted: Neste nível, uma transação pode ler dados não confirmados de outras transações. Isso significa que uma transação pode ver alterações feitas por outras transações que ainda não foram confirmadas (ou revertidas). Esse nível oferece o menor grau de isolamento e não é recomendado para ambientes de produção, pois pode resultar em leituras inconsistentes.

Read Committed: Neste nível, uma transação só pode ler dados que foram confirmados por outras transações. Isso evita leituras de dados não confirmados, mas ainda pode resultar em problemas de inconsistência, pois diferentes leituras dentro da mesma transação podem retornar resultados diferentes, caso outra transação tenha modificado os dados entre as leituras.

Repeatable Read: Neste nível, uma transação garante que todas as leituras dentro da transação retornem os mesmos resultados, mesmo que outras transações estejam modificando os dados ao mesmo tempo. Isso é alcançado por meio do bloqueio de leitura exclusiva em todas as linhas e tabelas que estão sendo lidas pela transação. No entanto, esse nível ainda permite a ocorrência de anomalias conhecidas como "phantom read", onde uma transação vê novas linhas que foram inseridas por outras transações após o início da transação atual.

Serializable: Esse é o nível mais alto de isolamento e garante que todas as transações ocorram como se fossem executadas sequencialmente, uma após a outra. Ele evita todas as anomalias de leitura, incluindo "phantom read". Isso é alcançado por meio de bloqueios de leitura e gravação exclusivos em todas as linhas e tabelas que são acessadas pela transação. No entanto, o nível de isolamento serializável pode resultar em bloqueios mais longos e pode afetar o desempenho em sistemas com grande concorrência.

Em suma, os níveis de isolamento permitem aos programadores controlar o equilíbrio entre consistência e desempenho, de uma base de dados relacional. Ao selecionar o nível adequado, obtem-se a segurança necessária para prevenir as eventuais anomalias de concorrência, prejudicando o mínimo possível a “performance” das transações.

O processo utilizado para a atribuição dos níveis de isolamento aos procedures, views e functions foi:

Deteção dos possíveis erros de concorrência nos mecanismos em questão e verificar, com base, nos níveis de deteção previamente explicados, qual o nível mais adequado perante as anomalias que poderiam ocorrer.

d) Todos os procedimentos presentes nesta alínea possuem apenas uma ação de escrita, logo podemos concluir que a única anomalia possível é o “Dirty write”.

Para proteger os procedures desta anomalia é apenas necessário o nível “read uncommitted”.

e) O procedimento presente nesta alínea possui leituras e escritas, uma das leituras é feita nos mesmos registos que a escrita, logo podemos concluir que as anomalias possíveis são “Dirty write” e “Dirty read”.

Para proteger o procedure desta anomalia é necessário o nível “read committed”.

f) O procedimento presente nesta alínea apenas possui leituras, logo podemos concluir que não é possível existirem anomalias de concorrência.

Para proteger o procedure é apenas necessário o nível “read uncommitted”.

g) O procedimento presente nesta alínea apenas possui leituras, logo podemos concluir que não é possível existirem anomalias de concorrência.

Para proteger o procedure é apenas necessário o nível “read uncommitted”.

h) O procedimento presente nesta alínea possui leituras e uma escrita, no entanto nenhuma leitura é realizada no mesmo local que a escrita, logo podemos concluir que a única anomalia possível é “Dirty write”.

Para proteger o procedure é apenas necessário o nível “read uncommitted”.

i) O procedimento presente nesta alínea possui leituras e escritas, uma das leituras é feita nos mesmos registos que a escrita, logo podemos concluir que as anomalias possíveis são “Dirty write” e “Dirty read”.

Para proteger o procedure desta anomalia é necessário o nível “read committed”.

j) O procedimento presente nesta alínea possui uma leitura e uma escrita, no entanto nenhuma leitura é realizada no mesmo local que a escrita, logo podemos concluir que não é possível existirem anomalias de concorrência.

Para proteger o procedure é apenas ncessário o nível “read uncommitted”.

k) O procedimento presente nesta alínea possui leituras e escritas, uma das leituras é feita nos mesmos registos que a escrita, logo podemos concluir que as anomalias possíveis são “Dirty write” e “Dirty read”.

Para proteger o procedure desta anomalia é necessário o nível “read committed”.

l) presente nesta alínea apenas possui uma leitura, logo podemos concluir que não é possível existirem anomalias de concorrência.

Para proteger o procedure é apenas necessário o nível “read uncommitted”.

m) O procedimento presente nesta alínea possui leituras e escritas, no entanto nenhuma leitura é realizada no mesmo local que a escrita, logo podemos concluir que a anomalia possível é “Dirty write”.

Para proteger o procedure é apenas ncessário o nível “read uncommitted”.

n) O procedimento presente nesta alínea possuem apenas uma ação de escrita, logo podemos concluir que a única anomalia possível é o “Dirty write”.

Para proteger os procedure desta anomalia é apenas necessário o nível “read uncommitted”.

# Avaliação Experimental

A avaliação da nossa solução é apresentada neste capítulo. Aqui mostramos

como as nossas grandes ideias funcionaram

Exemplo de indentação do segundo parágrafo.

## 4.1 Nome da primeira secção deste capítulo

Texto da secção.

Continuação do texto noutro parágrafo.

## 4.2 A segunda secção deste capítulo

Na segunda secção deste capítulo, vamos abordar o enquadramento, o contexto e as funcionalidades.

### 4.2.1 A primeira sub-secção desta secção

As sub-secções são úteis para mostrar determinados conteúdos de forma

organizada. Contudo, o seu uso excessivo também não contribui para a facilidade

de leitura do documento[[1]](#footnote-1).

### 4.2.2 A segunda sub-secção desta secção

Esta é a segunda sub-secção desta secção, a qual termina aqui.

## 4.3 Análise de resultados

A análise de resultados segue aqui, nos próximos parágrafos de forma detalhada.

A data limite de entrega da versão final em 19 de Setembro de 2015 tem subjacente a inscrição em época especial. Não se verificando esta situação, a data limite de entrega é em 25 de Julho de 2015. O júri de cada projecto é constituído por indicação do respectivo orientador, até 25 de Maio de 2015. A avaliação da versão beta será realizada até 29 de Junho de 2015.

A discussão da versão final do projecto é pública e inclui até 30 minutos de apresentação e demonstração seguidos de discussão até ao máximo de 120 minutos de duração total da prova (40 e 150 minutos, respectivamente, quando o grupo tem três estudantes, e, 20 e 90 minutos quando o trabalho é individual). O júri da prova é proposto pelo orientador do projecto, tem pelo menos três elementos e inclui o arguente, o orientador e um docente de Projecto e Seminário (podendo este, em caso de impedimento, delegar num docente da área departamental). As deliberações do júri são tomadas por maioria simples.

A melhoria de classificação, se realizada no semestre de Inverno seguinte, terá por base trabalho adicional e discussão oral avaliados pelo mesmo júri. Quando realizada noutro semestre, envolve a realização de novo projecto.

A coordenação global dos projectos e dos seminários é feita pelos docentes de Projecto e Seminário, de acordo com as orientações definidas pela comissão coordenadora do curso. No sítio desta unidade curricular, é mantida a informação relevante, incluindo prazos, calendário dos seminários, estado dos projectos e estudantes e orientadores envolvidos. No final de cada ano lectivo, o regente elabora e apresenta à comissão coordenadora do curso um relatório sucinto sobre o funcionamento da unidade curricular. Em cada ano lectivo são identificados os melhores projectos para promover a sua divulgação pública.

# Conclusões

Neste trabalho tratou-se o problema. Foi formulada a solução que assenta nos princípios de boas práticas aprendidos ao longo do curso.

A solução obtida atingiu resultados satisfatórios.

# Referências

|  |  |
| --- | --- |
| [1] | Wikipedia, “Big data --- Wikipedia, The Free Encyclopedia,” http://en.wikipedia.org/w/index.php?title=Big\_data&oldid=648786139, 2015. |
| [2] | X. Ding, X. Zhu e G. Wu, “Data mining with big data,” *IEEE Transactions on Knowledge and Data Engineering,* vol. 26, n.º 1, pp. 97-107, 2014. |
| [3] | J. Andrews, S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. Soong e J. Zhang, “What Will 5G Be?,” *IEEE Journal on Selected Areas in Communications,* vol. 32, n.º 6, pp. 1065-1082, 2014. |
| [4] | L. Boytsov, “Indexing Methods for Approximate Dictionary Searching: Comparative Analysis,” *J. Exp. Algorithmics,* vol. 16, n.º may, p. 1.81, 2011. |
| [5] | T. Jurkiewicz e K. Mehlhorn, “On a Model of Virtual Address Translation,” *J. Exp. Algorithmics,* vol. 19, n.º jan, pp. 1-18, 2015. |
| [6] | J. Neumann, The Computer and the Brain, New Haven, CT, USA: Yale University Press, 1958. |
| [7] | B. Kernighan e P. Plauger, The Elements of Programming Style, New York, NY, USA: McGraw-Hill, Inc., 1982. |

# A.1 Diagramas da Aplicação

Estamos no início do anexo 1. Nalguns casos, é conveniente colocar anexos de forma a complementar os resultados. Por vezes, em casos excepcionais devido à sua dimensão, as figuras têm que ser apresentadas de forma a ocupar toda a página, na forma de paisagem (*landscape*). Podemos fazer isso da forma que se apresenta na figura 3.



Figura 3 – Diagrama de casos de utilização.

# A.2 Modelos de dados

Estamos no início do anexo 2.

O *relatório* é um resumo do projecto global. Apenas como referência, é expectável cerca de 30 a 40 páginas A4 não devendo exceder 50 páginas.. A estrutura deve ser discutida e aceite pelo orientador. Os capítulos apresentados devem ter, em geral, a seguinte organização:

**Capítulo 1** — Introdução

Enquadramento do trabalho, metas, objectivos e especificações do projecto e resumo da solução. Concluir com a descrição breve dos restantes capítulos.

**Capítulo 2** — Formulação do problema

Introdução dos conhecimentos necessários para entendimento do trabalho, estabelecimento de terminologia e descrição detalhada do problema e do seu contexto. Síntese de abordagens anteriores do problema, caso existam, indicando as razões porque são insatisfatórias.

**Capítulo 3** — Grande ideia 1

Este capítulo pode ser subdividido em secções, designadamente:

1. Introdução: descrição da abordagem do problema e da metodologia utilizada; identificação das tarefas;
2. Elenco das características / Análise do problema: especificações, constrições, ferramenta de análise, etc*.*
3. Projecto: modelo para resolução do problema;
4. Implementação: a implementação do modelo como sistema computacional; descrição concisa do *hardware* e do *software*; dificuldades e contradições encontradas e sua resolução;
5. Avaliação: testes realizados e resultados experimentais (quando possível, o objectivo, a montagem e o método usado devem ser brevemente descritos); análise crítica dos resultados.

…

**Capítulo k+2** — Grande ideia k

**Capítulo k+3** — Conclusões

Recapitulação do trabalho desenvolvido. Referir claramente as observações e conclusões importantes. Discussão de ideias e recomendações para trabalho futuro.

**Referências**

Elenco dos livros e artigos citados no relatório. As referências são numeradas consecutivamente ao longo do relatório. O número da referência deve estar entre parêntesis rectos: [1].

**Anexos**

Os anexos devem incluir as partes importantes do *dossier do projecto*. O seu conteúdo depende da natureza do projecto, mas, em geral, pode incluir: listagem de programas, resultados de testes de *software*, exemplos de ecrãs de interface com o utilizador, esquemas dos circuitos, listagem de componentes, *data sheets* críticos, resultados de testes de *hardware*, desenhos mecânicos, análise económica, etc. (quando realizável, o *relatório* deve ser acompanhado da cópia do código, bibliotecas, etc. em suporte electrónico).

**Mais algumas dicas…**

O júri para avaliação do projecto final de curso apreciará o projecto, a sua demonstração e o respectivo relatório final (valorizando a escrita enquanto forma de divulgação de conhecimento). O relatório, depois de aceite e discutido, ficará disponível na biblioteca do departamento, para consulta.

O relatório deve ser auto-suficiente, isto é, o professor ou qualquer aluno finalista deve ficar apto a perceber o trabalho que realizou sem ter de ir à biblioteca ler os artigos originais.

Não escreva para o orientador, conhecedor de todo o detalhe, ou para um principiante. Tente escrever para uma audiência constituída por estudantes finalistas. Mantenha em mente o nível de conhecimentos do leitor a que se dirige. O relatório será uma base de trabalho para estudantes em circunstâncias semelhantes. Não dificulte o trabalho do leitor nem o faça estúpido (obviamente, …). Também é impossível ser totalmente claro. Evite afirmações dogmáticas (exemplo: "O *software* é a parte mais importante do computador.").

O relatório técnico não é uma história: usualmente não segue a cronologia das actividades realizadas. Também não é um romance (atenção à adjectivação). O relatório é um documento formal, feito para descrever os aspectos importantes do trabalho realizado.

Não tente descrever a função de cada componente, por exemplo a frase "O circuito IC2 e os componentes a ele associados formam um amplificador inversor …" é adequada. Contudo, descreva detalhadamente a função de cada componente ou circuito invulgar ou crítico.

As ilustrações (figuras, tabelas, gráficos e exemplos) são auxiliares preciosos para a explicação, mas envolvem muito trabalho. As figuras e as tabelas devem ser legíveis, instrutivas, legendadas e ter título. Os exemplos devem ser suficientemente detalhados para ilustrar o conceito.

O texto deve, pelo menos, ser analisado por um corrector ortográfico: os erros de ortografia são inadmissíveis.

Recomenda-se a leitura de alguns artigos e ou livros bem escritos para adquirir sensibilidade para a arte de escrever. Os artigos premiados em conferência são, normalmente, bons exemplos de escrita.

A escrita do relatório demora sempre mais tempo do que o inicialmente previsto.

**Resumo**

No essencial, a ideia é que tem algo para vender e o “Resumo” é a montra: a mensagem deve ser suficientemente clara e encorajar o cliente a entrar — se ele não a perceber passará ao lado. O resumo inclui: a motivação para o trabalho, como o fez e os resultados principais. Devem ser evitados chavões e palavras longas, as referências são proibidas e não deve utilizar acrónimos. Tenha em conta que o leitor será influenciado quer pela informação contida no resumo quer pela maneira como este está escrito. Não há desculpas para frases curtas ou desligadas, erros de gramática ou erros de sintaxe.

Não é fácil escrever um bom resumo.

**Introdução**

Procure dar resposta às seguintes questões: qual é o problema? porque é importante? o que é que outros já fizeram? quais as ideias base da solução apresentada? como está organizado o resto do relatório?

**Formulação do problema**

Defina o problema. Introduza a terminologia. Discuta as propriedades básicas.

**Solução do problema**

Enumere as coisas que fez e que considere importantes. Não seja modesto mas também não exagere.

A correcta avaliação do projecto é um aspecto crítico.

**Conclusões**

Procure dar resposta às seguintes questões: quais, se for caso disso, as lições aprendidas? o que, se algo, foi explicado? em que medida os objectivos foram atingidos? se existe algo que agora faria de forma diferente? quais as vantagens e desvantagens do trabalho realizado face a exemplos da literatura? que ideias para trabalho futuro?

**Referências**

A ideia subjacente à referência é que esta poupa papel e que o leitor poderá obter o documento em qualquer biblioteca científica razoável. Assim, é critério essencial referir revistas disponíveis em bibliotecas de instituições de ensino superior ou outras instituições profissionais. Em geral, não é razoável a referência a actas de conferências, dado que estas raramente estão acessíveis em bibliotecas pelo que, para todos os efeitos, estão indisponíveis. As referências a “Comunicações Privadas” são inaceitáveis. A informação dada deve ser sempre suficientemente detalhada por forma a que o leitor possa adquirir a publicação ou consultá-la numa biblioteca. Referências a dissertações de doutoramento ou outras devem indicar a instituição e o seu endereço. Sendo a referência essencial para o trabalho, no caso desta ser difícil de obter, dever-se-á incluir no documento, ou em apêndice, os seus pontos essenciais.

Cite uma referência sempre que está a incluir algo que não é contribuição sua ou quer indicar um conjunto de referências que o leitor pode consultar, mas cujo conteúdo não pode ser descrito adequadamente no relatório.

1. Este é um exemplo de nota de rodapé. Devem ser usadas com moderação, pois retiram a atenção ao texto principal. [↑](#footnote-ref-1)