Instituto Superior de Engenharia de Lisboa (ISEL)
Departamento de Engenharia Eletrónica e de Telecomunicações e Computadores (DEETC)

LEIM
Licenciatura em Engenharia Informática e Multimédia
Unidade Curricular de Projeto

Ferramenta de visualização de dados

(eventual) imagem ilustrativa do trabalho – *dimensão*: até 13cm x 4cm

Alice Fernandes 45741

Orientador(es)

Professor Paulo Trigo (ISEL)

Professor Artur (ISEL)

Professor Helder Fanha (ISCAL)

Junho, 2025

Resumo

Este relatório apresenta o desenvolvimento de uma ferramenta de visualização de dados, desenhada para apoiar os alunos que utilizam a plataforma *Marketplace Simulations - International Corporate Management*, durante o semestre. O principal objetivo da aplicação é simplificar a análise de dados, permitindo aos alunos carregar ficheiros Excel exportados da plataforma de simulação, processar e normalizar a informação carregada e mostrar gráficos interativos e dinâmicos.

A solução proposta introduz um modelo de dados baseado nos períodos de simulação da plataforma, em que os utilizadores podem gerir os dados. Cada ficheiro carregado é analisado e convertido em ficheiros CSV estruturados através de uma pipeline de normalização, construída utilizando *Python* e *Pandas*. Os conjuntos de dados resultantes são visualizados através de gráficos dinâmicos com recurso ao Plotly incorporados na interface gráfica *web*.

O projeto segue um design modular e centrado no utilizador, utilizando o *Django* como a biblioteca base da plataforma. As principais contribuições incluem a criação de um modelo de organização de ficheiros, automatismos no processamento de dados e uma interface gráfica intuitiva que vai de encontro às necessidades dos utilizadores. O sistema foi desenvolvido de forma incremental e está preparado para ser implementado num ambiente de produção, suportando vários utilizadores com isolamento de dados, controlo de versões.

Abstract

This report presents the development of a data visualization tool designed to support students using the Marketplace Simulations - International Corporate Management platform during the semester. The main aim of the application is to simplify data analysis by allowing users to upload Excel files exported from the simulation platform, automatically process and normalize the data and create interactive and dynamic graphs.

The proposed solution introduces a data model based on the platform's simulation periods, in which users can manage the data. Each uploaded file is analysed and converted into structured CSV files through a normalization pipeline built using *Python* and *Pandas*. The resulting data sets are visualized through dynamic graphs using Plotly incorporated into the *web* graphical interface.

The project follows a modular, user-centered design, using *Django* as the platform's base library. The main contributions include the creation of a file organization model, automated data processing and an intuitive graphical interface that meets users' needs. The system was developed incrementally and is ready to be implemented in a production environment, supporting multiple users with data isolation and version control.

Agradecimentos

Escrever aqui eventuais agradecimentos ...

Índice

Lista de Tabelas

Lista de Figuras

Lista de Acrónimos

API Application Programming Interface. 3

CSV Comma-Separated Values. 1

No sistema, o Comma-Separated Values (CSV) pe

Capítulo 1

Introdução

The Application Programming Interface (API) provides access to the hardware.

No contexto do ensino superior, a integração de ferramentas tecnológicas que potencializem o processo de aprendizagem é cada vez mais valorizada. Particularmente, em ambientes que simulam situações empresariais reais, como é o caso do simulador utilizado pelos alunos do ISCAL. A análise eficiente de dados torna-se essencial para a tomada de decisões estratégicas e tem impacto na avaliação final dos alunos, no entanto, a complexidade e a falta de visuais que ajudem a perceber as informações apresentadas pela plataforma original podem representar um desafio significativo para os estudantes. A

1.0.1 Motivação

O presente projeto surgiu da necessidade entre os alunos do ISCAL que utilizam o simulador empresarial *Marketplace Simulations – International Corporate Management* que é um simulador de negócios internacionais, onde os alunos se agrupam em empresas e simulam a criação de um negócio num mercado internacional. Embora a plataforma forneça toda a informação necessária à tomada de decisões na simulação, esses dados estão dispersos em múltiplas secções e apresentados em tabelas, sem funcionalidades de visualização gráfica ou filtragem. Esta limitação obriga os alunos a alternar entre páginas, copiar dados manualmente e criar folhas de cálculo externas, comprometendo tanto a eficiência quanto a qualidade da análise.

1.0.2 Objetivos

A aplicação proposta neste relatório pretende ajudar nesse sentido, oferecendo uma interface funcional que permite aos utilizadores carregar ficheiros XLSX exportados da plataforma originais e tornar esses ficheiros em visualizações que podem ser consultadas e manipuladas. A aplicação permitirá aos utilizadores:

- Criar uma conta na plataforma que permita persistir a informação carregada.
- Carregar ficheiros XLSX exportados da plataforma original;
- Visualizar os dados em gráficos interativos;

Do ponto de vista técnico, queremos que projeto adote uma arquitetura fácil ed manter e que vá de encontro à utilização da plataforma, dando ênfase aos seguintes itens:

- A normalização e transformação automática de dados provenientes de fontes externas;
- A facilidade na gestão de utilizadores e ficheiros, com o objetivo de oferecer uma experiência intuitiva para os utilizadores finais.
- Organizar a informação por utilizador, garantindo que o utilizador apenas consegue consultar a informação carregada.
- Adotar um modelo de organização semelhante à plataforma de simulação, de modo a tornar a experiência de utilização mais intuitiva e garantido que a nossa aplicação tenha fronteiras claras de utilização.

Ao longo deste relatório, serão detalhadamente apresentadas as decisões tomadas, bem como os fundamentos que orientaram o desenvolvimento da aplicação proposta.

Capítulo 2

Trabalho Relacionado

O presente projeto insere-se num contexto mais vasto de ferramentas pedagógicas e plataformas de apoio à tomada de decisão em ambientes simulados, particularmente no ensino superior com foco em gestão e estratégia empresarial. No âmbito do ISCAL, tem sido recorrente a utilização de plataformas como o *Marketplace Simulations*, que permitem aos estudantes desenvolver competências práticas em ambientes virtuais de negócios, replicando o funcionamento de mercados reais. Esta necessidade de suporte digital à análise e interpretação de dados já motivou o desenvolvimento de outras ferramentas auxiliares, com destaque para um projeto também realizado em parceria com o ISCAL, focado em simulações parciais de modelos económicos simplificados.

Esse projeto, embora partilhe a mesma motivação — facilitar a análise de informação extraída de simuladores empresariais — apresentava uma abordagem distinta. Em particular, a aplicação permitia simular cenários específicos com base em inputs manuais ou em pequenos ficheiros com dados sintetizados, o que se revelou útil para exercícios de curto alcance ou com foco muito delimitado (por exemplo, simular o impacto de uma única variável nos resultados de uma empresa virtual). No entanto, não oferecia suporte direto ao processamento automático dos dados reais exportados da plataforma *Marketplace Simulations*, nem dispunha de funcionalidades integradas de normalização de dados ou visualização gráfica dinâmica.

O projeto atual distingue-se, assim, tanto pelo seu alcance como pela profundidade técnica das soluções propostas. Ao contrário do sistema anterior, que se baseava em simulações parametrizadas e modelos estáticos, esta nova

aplicação aposta numa abordagem orientada a dados, onde a informação real dos jogos de simulação é carregada diretamente pelo utilizador. Essa escolha trouxe consigo a necessidade de resolver questões de normalização, estruturação e tratamento de grandes volumes de folhas de cálculo, algo que não era exigido nos projetos anteriores.

Além disso, o trabalho atual introduz um modelo de organização baseado em "quarters", associando cada conjunto de dados a um ciclo temporal definido e mantendo a rastreabilidade por utilizador. A utilização de um pipeline modular de limpeza e conversão de dados, aliado à geração automática de gráficos interativos com filtros avançados, representa um avanço significativo em termos de sofisticação funcional. Também ao nível da arquitetura, a adoção de tecnologias como Django, Pandas, Plotly e WebComponents configura uma base técnica robusta, capaz de escalar e ser adaptada a novos contextos.

Este trabalho assume, portanto, como pressuposto o uso de dados reais extraídos diretamente do simulador, a necessidade de integração com um sistema de autenticação e gestão de utilizadores, e a valorização da experiência do utilizador (UX) na apresentação de dados. Em conjunto, estes elementos definem uma solução mais abrangente e alinhada com os desafios técnicos reais da análise de dados em contexto educativo, indo além das simulações reduzidas ou controladas que caracterizavam os trabalhos relacionados anteriormente desenvolvidos.

Capítulo 3

Modelo Proposto

Aqui mostra um caminho que inicia com requisitos (cf., secção 3.1), passa pela aplicação dos fundamentos (cf., secção 3.2) e continua até conseguir transmitir uma visão clara e um formalismo com nível de detalhe adequado a um leitor que tenha um perfil (competência técnica) idêntico ao seu.

Recorra, sempre que possível, a exemplos ilustrativos da utilização do seu modelo. Esses exemplos devem ajudar o leitor a compreender os aspectos mais específicos do seu trabalho.

O modelo aqui proposto deve ser (tanto quanto possível) independente de tecnologias concretas (e.g., linguagens de programação ou bibliotecas). No entanto,, deve fornecer os argumentos que contribuam para justificar uma posterior escolha (adoção) de tecnologias.

(falta introduzir o modelo proposto)

3.1 Requisitos

Aqui o essencial (e se aplicável) dos requisitos funcionais, não funcionais e modelo de casos de utilização. Aqui deve também apresentar matriz para decisão sobre prioridade dos casos de utilização (se aplicável) ...

Deve apresentar de forma "moderada" o resultado da fase avaliação de requisitos. A informação de maior detalhe (e.g., diagramas UML demasiado detalhados) deve ser colocada em apêndice.

Requisitos funcionais

Os requisitos funcionais descrevem as funcionalidades específicas que a aplicação deve oferecer para atender às necessidades dos utilizadores. No contexto deste projeto, definem as ações que o sistema deve ser capaz de executar. No nosso projeto, identificamos os seguintes requisitos:

- **Visualizações dinâmicas:** a aplicação deve conseguir mostrar gráficos com base nos dados carregados, com suporte a alteração de parâmetros em tempo real (por exemplo, mudar de *quarter* ou selecionar um país específico).
- **Upload e processamento de ficheiros XLSX:** os utilizadores devem conseguir carregar ficheiros *Excel* com múltiplas folhas; esses ficheiros são automaticamente convertidos para CSV e normalizados.
- **Autenticação e gestão de utilizadores:** cada utilizador tem uma conta e pode gerir os seus próprios dados. Apenas os ficheiros mais recentes serão considerados para as visualizações.
- **Gestão de *quarters*:** os utilizadores podem criar períodos temporais (*quarters*), cada um identificado por um número, que funcionam como *bucket* lógicos para organizar os ficheiros carregados.

Requisitos não funcionais

Alguns requisitos não funcionais foram igualmente críticos para garantir a robustez e usabilidade do sistema (TODO):

- **Normalização da informação:** A *pipeline* de processamento é isolada e modular, facilitando a manutenção e futura extensão do sistema.
- **Normalização dos dados:** foram aplicadas rotinas automáticas para garantir coerência nos nomes de colunas, remoção de quebras de linha, e eliminação de colunas irrelevantes.
- **Isolamento por utilizador:** cada utilizador só pode aceder aos seus dados, e visualizar a informação que carregou.

- **Experiência de utilizador:** A plataforma tem de ser capaz de oferecer uma boa experiência de utilização.

(Colocar as actual tabelas no apendice)

3.1.1 Casos de Utilização

(TODO)

- Colocar UML commo apendice - Referenciar a matriz prioridade dos casos de utilização Com os casos de utilização estabelecidos, foram identificados então os requisitos da plataforma, que podemos separar em requisitos funcionais e não funcionais

Falta a matriz de prioridade dos casos de utilização

3.2 Fundamentos

A aplicação desenvolvida segue a mesma organização que já existe na plataforma *Marketplace Simulations*, por *quarters*, em que cada *quarter* corresponde a uma semana simulada, permitindo que cada utilizador possa carregar ficheiros Excel extraídos da plataforma . Esses ficheiros são tratados e depois transformados em CSV, e usados para criar visualizações.

Significa que, estruturalmente, a aplicação depende de três entidades diferentes, que iremos descrever abaixo, e como se interligam no funcionamento da aplicação

3.2.1 *Quarters*

Os *quarters* funcionam como *buckets* lógicos para organizar os ficheiros carregados pelos utilizadores. Cada utilizador pode criar múltiplos *quarters*, identificados de forma única por um número. Este número serve tambem de identificador, sendo que não é possível ter dois *quarters* identificados como 1 para cada utilizador.

A nível de implementação, cada *quarter* está associado unicamente a um utilizador e é identificado por um UUID (gerado automaticamente pelo *Django*), que garante que o *quarter* seja único.

3.2.2 Ficheiros

Os ficheiros são inicialmente carregados no formato Excel, contendo uma ou várias folhas de cálculo. Cada folha é tratada como uma entidade individual e transformada para CSV. O ficheiro Excel é guardado como referência, mas não é diretamente utilizado para visualização.

Este processo de conversão para CSV é acompanhado por uma *pipeline* de normalização de dados, que limpa os dados (como por exemplo, remover quebras de linha, colunas sem representação, nomes inconsistentes) e aplica regras para que os gráficos possam ser gerados de forma consistente. Cada ficheiro CSV gerado é associado ao seu ficheiro Excel de origem, ao *quarter* correspondente, e o seu nome será baseado no nome da folha de onde foi extraído. Aos ficheiros CSV é também associado uma *slug*, baseado também no nome, que identifica a informação que o CSV representa.

A plataforma garante que só existe uma versão ativa de cada ficheiro por tipo — caso o utilizador carregue novamente um ficheiro com o mesmo nome lógico, o anterior será marcado como não ativo, evitando duplicações e garantindo que os gráficos usam apenas dados mais recentes.

3.2.3 Utilizadores

A plataforma foi desenhada para funcionar com utilizadores, baseando-se no sistema de autenticação pré-existende do *Django*. Cada utilizador tem a sua conta, e pode realizar operações como criação de *quarters*, carregamento e alteração de ficheiros, e aceder aos gráficos gerados a partir desses ficheiros.

Cada utilizador tem acesso apenas aos seus próprios dados, garantindo o isolamento da informação. Esta separação é feita a nível da base de dados, através da associação de cada entidade ao utilizador que criou.

Apesar da plataforma não suportar explicitamente equipas ou grupos, assume-se que alunos do mesmo grupo podem carregar ficheiros semelhantes, mas o sistema trata-os como ficheiros diferentes. Assim, evita-se a complexidade adicional de gerir permissões ou partilha de dados entre contas. Também se assume que as contas podem ser criadas ao nível do grupo, pelo que para a plataforma, é indiferente se a conta é individual ou partilhada entre membros desse grupo.

No futuro, pode ser considerada a funcionalidade de desativação au-

tomática de contas (por exemplo, após o final do semestre), mas para já o modelo é simples e robusto: conta individual, dados isolados, e controlo completo sobre os próprios uploads.

3.3 Abordagem

Para a concretização do projeto, definimos então algumas abordagens que afetam a maneira como a plataforma é desenvolvida e utilizada. Apesar de o produto final ser uma aplicação *web*, existem fatores diferenciadores neste projeto que só conseguem ser explicados em contexto com o problema que queremos resolver.

3.3.1 Organização por *Quarters*

Uma das primeiras decisões que foram tomadas foi como organizar a informação recebida, e para facilitar a utilização da plataforma, foi decidido organizar os dados por *quarters*, refletindo o modelo temporal da simulação, onde as decisões são tomadas em em *quarters*. Cada *quarter* corresponde a uma semana, e a cada *quarter* é possivel exportar um conjunto de ficheiros da plataforma original. Assim conseguimos garantir que a interface visual é intuitiva e fácil de perceber, uma vez que o mapeamento dos *quarters* é igual no modelo proposto.

Esta organização é feita de forma intencional, pelos utilizadores da plataforma, ao criarem *quarters* na plataforma, onde posteriormente carregam a informação. Os *quarters* ajudam também a segmentar a informação carregada, e indica explicitamente à plataforma onde pertencem os dados.

A alternativa seria inferir o *quarter* através do nome do ficheiro (como por exemplo, *CustomerNeeds-Q5.xlsx*, mas esta alternativa assumia que os alunos não trocam o nome ou não organizam os ficheiros de outra maneira e faria com que a plataforma dependesse de um identificador (nome do ficheiro) externo para determinar onde associar o ficheiro recebido.

3.3.2 *Pipeline* de Normalização de Dados

Outro contributo diferenciador foi a criação de uma *pipeline* modular de normalização de dados. O objetivo é garantir que os ficheiros Excel carregados,

que muitas vezes contêm nomes de colunas inconsistentes, quebras de linha, espaços em excesso ou colunas irrelevantes, sejam convertidos em CSV com um formato adaptado para visualização.

Como podemos receber muitos ficheiros, a variabilidade entre os dados recebidos é muito alta, pelo que alguns dados passam por mais do que uma fase de normalização. Esta decisão foi tomada com base numa análise manual, em que identificámos possíveis fontes de dados que precisam de mais do que uma fase de normalização. As várias fases de normalização alteram os dados de modo a facilitar a representação visual dos mesmos e é um passo essencial no projeto, porque garante que a aplicação trabalha com formatos e regras conhecidas, e remove a variabilidade dos ficheiros importados.

As fases de normalização irão ser descritas em mais detalhe nos capítulos seguintes, uma vez que a implementação destas pipeline estão relacionadas à tecnologia escolhida, mas o desenvolvimento desta *pipeline* é um fator diferenciador deste projeto, uma vez que tem de lidar com dados que não estão estruturados de forma a facilitar representações visuais.

Este processo de normalização é semelhante aos processos ETL (Extract, Transform, Load) ainda que neste projeto tenha sido desenvolvido com uma escala menor.

Capítulo 4

Implementação do Modelo

O desenvolvimento da aplicação assentou em fundamentos sólidos tanto ao nível tecnológico como ao nível das boas práticas de engenharia de software e tratamento de dados. A escolha da arquitetura, das tecnologias utilizadas e dos modelos de organização dos dados foi orientada por critérios de fiabilidade, modularidade, simplicidade de manutenção e escalabilidade.

4.1 Arquitetura Tecnológica

- Falar sobre *stack* usada no projeto, alternativas e prós e contras

4.2 Análise dos dados

Nesta secção iremos descrever como analisamos a informação recebida da plataforma de simulação e as decisões tomadas sobre essa informação.

4.2.1 Análise inicial da informação

Após termos recebido os dados, fizemos uma primeira análise com o objetivo de perceber a informação recebida e os próximos passos a tomar. No total, recebemos 39 ficheiros Excel que foram exportados da plataforma,

4.2.2 Classificação da informação

- Falar sobre como classificamos os 39 ficheiros Excell em "buckets"e cada bucket ficou com um "template"gráfico pré-determinado - Falar sobre cada

grupo, o que significa e o que todos os gráficos partilham em comum em cada grupo.

4.2.3 Visualização escolhidas para os vários grupos

- Falar sobre os vários

4.3 Processamento dos dados

- Iniciar o tema de processamento de informação, converter informação desconhecida para um formato normalizado.

4.3.1 *Pipeline* de transformação dos dados

A primeira fase de normalização segue os seguintes passos:

- Extrai o nome do gráfico a partir da primeira linha de cada folha.
- Usa a segunda linha como cabeçalhos e normaliza os nomes (ex: substitui espaços por *underscores*, remove quebras de linha).
- Remove colunas irrelevantes com base numa lista de columnas que não tem representação.
- Normaliza dados nas células (como por exemolo: remove quebras de linha, espaços duplos).

A segunda fase vai depender do ficheiro e do gráfico que queremos apresentar, mas no geral,

(...) completar isto

- Falar como foi desenvolvida, e as várias fases de transformação A desenvolvida em *Python* com recurso ao Pandas.

4.3.2 Conversão e Exportação para CSV

- Falar sobre a transformação do pandas- Falar da exportação do CSV- Falar da gestão de ficheiros CSV e Excel, e metodologias para não haver conflito entre ficheiros carregados

4.4 Desenvolvimento da Aplicação Web

Esta secção foca-se na estrutura e funcionamento da aplicação web — tanto o backend em Django como a interface construída com HTML, WebComponents e Flowbite.

4.4.1 Arquitetura da aplicação Django

- Descrever como está organizada a aplicação: os modelos principais (Quarter, ExcelFile, CSVFile), as relações entre eles e como são usados para garantir o isolamento por utilizador.
- Explicar o sistema de autenticação (Django Auth) e como se garantem as permissões e o acesso aos dados por utilizador.
- Referir também o uso do sistema de media (uploads) com UUIDs por quarter, e a criação automática das pastas.

4.4.2 Endpoints e API interna

Detalhar os principais endpoints utilizados: uploads, visualização de gráficos, listagem de ficheiros, etc.

- Mencionar a estrutura REST dos endpoints e como a interface os consome (por exemplo, o /api/charts/¡chart$_id$ > / < $quarter$ > /).
-
- Referir como se gere o estado do sistema com propriedades como $is_current, eoqueacontecequando$

4.4.3 Sistema de processamento assíncrono e normalização

- Falar sobre o $run_pipeline_for_sheet(...)dodata_processing.pyecomoissoseencaixacomomodeloExce$
-

- *Comentar o sistema de marcação dos ficheiros como processados e o uso de flags como processed.*

4.5 Interface Gráfica e Experiência do Utilizador (Frontend)

Aqui explicas como a interface foi pensada, as ferramentas utilizadas e como os gráficos são construídos com base nos dados enviados pelo backend.

4.5.1 Design System e Flowbite

Explicar por que escolheste Flowbite e como ele ajudou a construir uma UI consistente e reutilizável.

Mostrar exemplos de componentes reutilizados, como modais, botões, tabs, etc.

4.5.2 WebComponents e gráficos interativos

Falar sobre como encapsulaste a lógica dos gráficos usando WebComponents, para evitar conflitos de JS e garantir modularidade.

Descrever a comunicação entre os WebComponents e o backend via fetch, passando parâmetros (por exemplo, o quarter atual, ou filtros).

4.5.3 Gestão de Quarters e Uploads

Mostrar como funciona a criação de quarters e a navegação entre diferentes trimestres (com os botões e setas).

Explicar o fluxo de upload de ficheiros e como a interface valida o tipo de ficheiro, evita duplicações, e atualiza a visualização após carregamento.

Capítulo 5

Validação e Testes

Validação e testes aqui ...; pode precisar de referir o capítulo 3 ou alguma das suas secções, e.g., a secção 3.2 ...

Pode precisar de apresentar tabelas. Por exemplo, a tabela 5.1 apresenta os dados obtidos na experiência ...

c_1	c_2	c_3	$\sum_{i=1} c_i$
1	2	3	6
1.1	2.2	3.3	6.6

Tabela 5.1: Uma tabela

Para além de tabelas pode também precisar de apresentar figuras. Por exemplo, a figura 5.1 descreve ...

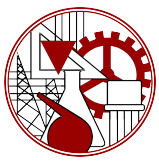

Figura 5.1: Uma figura

Atenção. Todas as tabelas e figuras, e.g., diagramas, imagens ilustrativas da aplicação em funcionamento, têm que ser devidamente enquadradas no texto antes de serem apresentadas e esse enquadramento inclui uma explicação da imagem apresentada e eventuais conclusões (interpretações) a tirar dessa imagem.

Capítulo 6

Conclusões e Trabalho Futuro

Conclusões e trabalho futuro aqui ...

Quais as principais mensagens a transmitir ao leitor deste trabalho? O leitor está certamente interessado nos temas aqui abordados. Em geral procurará, neste projeto, pistas para algum outro objetivo. Assim, é muito importante que o leitor perceba rapidamente a relação entre este trabalho e o seu próprio (do leitor) objetivo.

Aqui é o local próprio para condensar a experiência adquirida neste projeto e apresentá-la a outros (futuros leitores).

O pressuposto é o de que de que este projeto é um "elemento vivo" que recorreu a outros elementos (cf., capítulo 2) para ser construído e que poderá servir de suporte à construção de futuros projetos.

Apêndice A

Tutorial

O "apêndice" utiliza-se para descrever aspectos que tendo sido desenvolvidos pelo autor constituem um complemento ao que já foi apresentado no corpo principal do documento.

Neste documento utilize o apêndice para explicar o processo usado na **gestão das versões** que foram sendo construídas ao longo do desenvolvimento do trabalho.

É especialmente importante explicar o objetivo de cada ramo ("branch") definido no projeto (ou apenas dos ramos mais importantes) e indicar quais os ramos que participaram numa junção ("merge").

É também importante explicar qual a arquitetura usada para interligar os vários repositórios (e.g., Git, GitHub, DropBox, GoogleDrive) que contêm as várias versões (e respetivos ramos) do projeto.

Notar a diferença essencial entre "apêndice" e "anexo". O "apêndice" é um texto (ou documento) que descreve trabalho desenvolvido pelo autor (e.g., do relatório, monografia, tese). O "anexo" é um texto (ou documento) sobre trabalho que não foi desenvolvido pelo autor.

Para simplificar vamos apenas considerar a noção de "apêndice". No entanto, pode sempre adicionar os anexos que entender como adequados.

Apêndice B

Casos de Utilização - UML

Apêndice C

Tabela de Requisitos funcionais

Apêndice D

Tabela de Requisitos não funcionais

Apêndice E

Classificação das folhas Excel