Relatório de Especificação: Desenho

[Os comentários e algum conteúdo exemplificativo incluídos no documento destinam-se a apoiar na sua preparação e estão indicados a verde. **Remover os comentários na versão para entrega**.

Os *templates* dos relatórios fornecem uma abordagem consistente para organizar as informações de um projeto. Ajudam a lembrar os conteúdos que é necessário preparar. Não é preciso que se preencha o *template* sequencialmente, do início ao fim. Se determinadas seções de um *template* não são realmente necessárias, deve-se removê-las.]

|  |  |
| --- | --- |
| Projeto: | <nome do projeto aqui> |
| Preparado por: | Grupo xxxx  <nome aluno> (nº 999999)  <nome aluno> (nº 999999)  [ lista dos membros que realizaram (efetivamente) o trabalho] |
| Data de preparação: | Aveiro, 31 de fevereiro de 2017 |
| Circulação: | Docentes e Discentes de AMS. |

Controlo de versões

|  |  |  |
| --- | --- | --- |
| Quando? | Responsável | Alterações significativas |
| <data> | <quem alterou>? | <descrição das principais alterações/secções introduzidas> |
|  |  |  |
|  |  |  |

Índice

[1 Introdução 4](#_Toc24842696)

[2 Objetivos e restrições da arquitetura 7](#_Toc24842697)

[2.1 Objetivos da arquitetura face aos requisitos do domínio 7](#_Toc24842698)

[2.2 Atributos de qualidade com impacto na arquitetura 7](#_Toc24842699)

[2.3 Decisões tomadas e justificação 8](#_Toc24842700)

[3 Vistas de arquitetura 9](#_Toc24842701)

[3.1 Arquitetura lógica da solução 9](#_Toc24842702)

[3.2 Arquitetura operacional 9](#_Toc24842703)

[4 Modelo de comportamento 10](#_Toc24842704)

[4.1 Interação entre módulos 10](#_Toc24842705)

[4.2 Operações de sistema 10](#_Toc24842706)

[4.3 Integrações com sistemas externos 10](#_Toc24842707)

[5 Tecnologias e implementação 11](#_Toc24842708)

[6 Protótipos dos produtos 12](#_Toc24842709)

[7 Anexos 13](#_Toc24842710)

# Introdução

[Introdução ao documento: qual é o assunto do documento e porque o estamos a fazer?

Resumo dos principais resultados que podem ser apreciados no documento. ]

Índice de diagramas

[actualizar o índice quando o documento completo. Todos os diagramas devem ser legendados.]

[Diagrama 3: Modelo do domínio. **Error! Bookmark not defined.**](#_Toc417508552)

Índice de tabelas

[atualizar o índice quando o documento completo. Todas as tabelas devem ser legendadas.]

[Tabela 1: Principais benefícios do CourseRegistrarSystem. **Error! Bookmark not defined.**](#_Toc214204493)

# Objetivos e restrições da arquitetura

## Objetivos da arquitetura face aos requisitos do domínio

[Identificar, de uma forma geral, as caraterísticas críticas do problema/solução que vão determinar as grandes decisões da arquitetura.

*Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues?*

*Are there hardware dependencies that should be isolated from the rest of the system?*

*Does the system need to function efficiently under unusual conditions?*

*Are there integrations with external systems?*

*Is the system to be offered in different user-interfacing platforms (web, mobile devices, big screens,…)?*

Are there special requirements concerning the extent or the type of the data managed (e.g.: large collection of videos)?

Etc.]

E.g.:

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

* The existing legacy Course Catalog System at Wylie College will be the main source for all course information; it must be accesses to retrieve the course details for the current semester. The C-Registration System must support the data formats and DBMS of the legacy Course Catalog System [see Report X].
* The existing legacy Billing System at Wylie College must be interfaced with to support billing of students. This interface is defined in the Course Billing Interface Specification [see Report X].
* The C-Registration System will be implemented as a scalable system, deployed in the cloud. The users should access using only their browser, with no required configuration, to the web portal.

## Atributos de qualidade com impacto na arquitetura

[Enumerar atributos de qualidade (naturalmente já formulados como requisitos não funcionais em relatórios anteriores) com impacto nas escolhas relativas à arquitetura. Só incluir os que contribuem para influenciar a arquitetura do sistema.

Esta secção é diferente da anterior. Na anterior, trata-se de restrições gerais; nesta secção, a ideia é fazer um “apanhado” de requisitos, formulados como tal (S.M.A.R.T.), já identificados antes]

E.g.:

* + 1. The desktop user-interface shall be Windows 95/98 compliant.
    2. The user interface of the C-Registration System shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System.
    3. Each feature of the C-Registration System shall have built-in online help for the user. Online Help shall include step by step instructions on using the System. Online Help shall include definitions for terms and acronyms.
    4. The C-Registration System shall be available 24 hours a day, 7 days a week. There shall be no more than 4% down time.
    5. Mean Time Between Failures shall exceed 300 hours.
    6. Upgrades to the PC client portion of C-Registration shall be downloadable from the UNIX Server over the internet. This feature enables students to have easy access to system upgrades.

## Decisões tomadas e justificação

[List the decisions that have been made regarding architectural approaches and the constraints being placed on the way that the developers build the system.

These will serve as guidelines for defining architecturally significant parts of the system. Justify each decision or constraint so that developers understand the importance of building the system according to the context created by those decisions and constraints. This may include a list of DOs and DON’Ts to guide the developers in building the system.]

E.g.:

- **compatibilidade com OneM2M**: a camada de IoT será baseada na utilização de *middleware* já existente, para implementar a compatibilidade com o stardard OneM2M. A implementação *opensource* do Eclipse OneM2M será usada como referência. A versão do *middleware* não poderá ser revista a meio de uma iteração.

# Vistas de arquitetura

## Arquitetura lógica da solução

[Como é que a solução de software vai estar organizada?

Explicar a organização prevista do software em módulos, camadas ou subsistemas. Para além dos diagrama, é necessário texto de explicação.

Sugestão: apresentar uma arquitetura lógica por camadas (layered architecture), visualizada com um diagrama de pacotes]

Na descrição da arquitetura, explicar as Tecnologias e ambientes e suporte que se prevê utilizar: web, mobile,…

## Arquitetura operacional

[Explicar a organização prevista da solução em termos configuração de produção (*deployment*). Modelar num diagrama de *deployment*]

# Modelo de comportamento

## Interação entre módulos

[Tomando como referência os módulos identificados no ponto 3.1, explicar as principais interações que acontecem entre os módulos, a alto nível. Podem ser usados D. de Sequência para mostrar a interação. Não é preciso ser exaustivo, mas deve ficar claro, para um programador que integre a equipa, como é que as partes definidas na arquitetura vão comunicar/colaborar.]

## Operações de sistema

[utilizar Diagramas de Sequência de Sistema para identificar as operações de sistema. Nota: como explorado nas aulas TP, as operações se sistema identificam os pontos de entrada no sistema, a partir dos fluxos dos casos de utilização.

Recomendação: pode-se fazer apenas para um subconjunto dos casos de utilização, justificando que o subconjunto considerado diz respeito aos casos de utilização a implementar na iteração imediata]

## Integrações com sistemas externos

[explicar as colaborações entre o nosso e outros sistemas de software, identificando (a interação com) a **troca de mensagens**., visualizada com diagramas de sequência]

# Tecnologias e implementação

[Explicar as tecnologias selecionadas para a implementação: linguagens de programação, *frameworks* de que se vai tirar partido, sistemas de comunicação a usar, protocolos para a integração de sensores e dispositivos, que sensores/devices foram escolhidos, etc.]

[Explicar as atividades de validação da arquitetura que já foram desenvolvidas.

Referir, se aplicável, os recursos relevantes relacionados com os projetos feitos para validar a arquitetura proposta (e.g.: repositório git de código partilhado)]

# Protótipos dos produtos

[maquetas gerais dos produtos de software. Não só protótipos funcionais.

A ideia não é mostrar todos os *screens*, mas incluir ilustrações representativas das opções tomadas

Deve ser acompanhado com referências/links para versões experimentáveis dos protótipos]

[maquetas de outros *outputs*, *eg*: mapas estatísticos a produzir, painéis de apoio à decisão, etc.]

# Anexos

[listar a documentação anexa, referida ao longo do relatório]

[listar ficheiros externos, nomeadamente versões eletrónicas do relatório, modelos UML a consultar ou sítios com as maquetas]

[anexar cópias de documentos chave relevantes para a documentação de requisitos, e.g., formulários]