INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




SWiPE - Sistema de Informação Configurável e Auditável para Gestão das UC de Projeto Curricular do ISEP

JOANA FILIPA FERREIRA CARNEIRO DOS SANTOS
Outubro de 2019

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

SWiPE
Sistema de Informação Configurável e Auditável para Gestão das UC de Projeto Curricular do ISEP

Joana Filipa Ferreira Carneiro dos Santos

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software

Orientador: Dr. Nuno Alexandre Pinto da Silva

Porto, outubro de 2019

Dedico esta tese ao meu avô José Fernando, que apesar de já não estar comigo, acredito que teria o maior orgulho em todo este percurso.

Resumo

No ISEP, todos os cursos de mestrado e a maioria das licenciaturas possui no seu plano de estudos uma unidade curricular (UC) de projeto-estágio. Cada uma destas UC tem o seu funcionamento próprio e as suas fases específicas, tornado assim muito complexo o processo de gestão. O facto do número de estudantes a realizar estas UC aumentar de ano para ano, traduz-se também num aumento da complexidade e esforço deste processo.

Aliado a isto, o facto de não existir software integrado que permita auxiliar este processo, torna relevante o desenvolvimento de um sistema que permita dar resposta aos diferentes processos de gestão, permitindo reduzir o número e esforço das etapas manuais existentes.

Através de várias atividades de elicitação de requisitos, identificaram-se as especificidades das diversas UC e sistematizaram-se processos e etapas comuns, o que permitiu concluir da necessidade de adaptabilidade (identificar pontos de evolução e permitir a evolução do sistema nesses pontos/necessidades/funcionalidades) e configurabilidade (que pressupõe permitir a parametrização do funcionamento específico de cada edição de UC, portanto, alterável ao longo do tempo).

Desenvolveu-se, então um software que suporte aos diferentes processos de gestão das várias UC, cumprindo a legislação e os regulamentos internos aplicáveis. Para tal, elaborou-se o design arquitetural do sistema e das suas partes, adotando racionais suportados por princípios, estilos e padrões existentes na literatura.

Conduzido pelo design arquitetural proposto, procedeu-se à implementação da aplicação servidora capaz de gerir informação de configuração do funcionamento das UC, e de gerir a informação de operação (estruturada pela informação de configuração). Complementarmente, construiu-se uma aplicação cliente web do tipo *Single Page Application*, que fornece aos atores do sistema interface e funcionalidades específicas consoante a informação de configuração da UC disponibilizada pela aplicação servidora.

Os diversos testes automáticos (i.e. unitários, de integração, de sistema e de aceitação) desenvolvidos permitem afirmar que a abordagem arquitetural e em particular a separação de funcionalidades de configuração e operação, cumpre os requisitos funcionais das UC.

As experiências de desempenho realizadas permitem concluir que o sistema cumpre as métricas requeridas, mesmo em situações (simuladas) de picos de uso.

Palavras-chave: ISEP, projeto-estágio, heterogeneidade, configurabilidade

Abstract

At ISEP, all master's courses and most undergraduate degrees don't have their syllabus for a subject. Each of these subjects has its own operation and specific phases, made as complex as the management process. The fact, that the number of students taking these subject increases every year also translates into increased complexity and effort of this process.

Allied to this, the fact that there is no integrated software that allows this auxiliary process, becomes relevant, the development of a system that allows to respond to different management processes, allowing to reduce the number and perform the necessary steps.

Through various requirements elicitation activities, identified as specific to several subjects and systematized common processes and steps, or that recover from the need for adaptability (identify evolution points and allow the system evolution of these points/requirements/functionalities) and configurability (which presupposes allowing a parameterization of the specific functioning of each subject, therefore, changeable over time).

Then develop software that supports the different management processes of various subjects, complying with the laws and regulations used. To do this, prepare an architectural design of the system and its parts, adopting the requirements applicable to the principles, styles and standards in the literature.

Conducted by the proposed architectural design, the server application was implemented to manage the configuration information of the subject operation, and to manage the operation information (structured by the configuration information). In addition, a Single Page Application web client application was built, which provides system actors with specific interface and functionality depending on the subject configuration information provided by the server application.

The various automated tests (e.g. units, integration, system, and acceptance) can apply the effects of the architectural approach and, in particular, the varying configuration and operation configurations, meet UC functional requirements.

The performance experiments performed conclude the system meets the required measurements, even in (simulated) peak usage situations.

Keywords: ISEP, intership, heterogeneity, configurability

Agradecimentos

Em primeiro lugar, agradeço ao Instituto Superior de Engenharia do Porto a confiança depositada na produção deste *software* e a todos os docentes que de alguma forma participaram na realização deste projeto: os que transmitiram os seus conhecimentos ao longo destes dois anos do Mestrado em Engenharia Informática e aqueles que disponibilizaram um pouco do seu tempo para a realização de reuniões e que emitiram as suas opiniões acerca do *software* desenvolvido.

Agradeço ao Doutor Nuno Silva por todo o apoio e orientação prestados ao longo do projeto, salientando a confiança depositada na realização deste *software*, todo o tempo despendido na discussão de progressos e na revisão do trabalho produzido, bem como todas as críticas construtivas e sugestões que permitiram obter um trabalho com muito melhor qualidade.

Deixo um agradecimento a todos os colegas de curso com os quais tive a oportunidade de partilhar os momentos de estudo e trabalho, mas também os de lazer e que de certa forma me ajudaram a atingir este objetivo.

Por fim, porque os últimos são sempre os primeiros, gostaria de agradecer aos meus pais, por me proporcionarem as melhores condições tanto a nível pessoal como académico, o que me permitiu concluir esta etapa universitária. À minha avó Ermelinda, deixo o meu mais especial agradecimento pelo apoio incondicional ao longo da realização deste projeto e por ter sempre a frase certa para o momento.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos & Siglas

AHP	*Analytic Hierarchy Process*
API	*Application Programming Interface*
CD	*Continuous Delivery*
CI	*Continuous Integration*
CRUD	*Create, Read, Update and Delete*
DIP	*Dependency Inversion Principle*
DIPRE	Dissertação/Projeto/Estágio do Mestrado em Engenharia Civil
DISEST	Dissertação/Estágio do Mestrado em Engenharia Química
DRY	*Don't Repeat Yourself*
ESTBI	Estágio da Licenciatura em Biorrecursos
FFE	*Fuzzy Front-End*
FURPS	Funcionalidade, Usabilidade, Confiabilidade, Performance e Suportabilidade
GRASP	*General responsibility assignment software patterns*
HATEOAS	Hypermedia as the Engine of Application State
HTML	*HyperText Markup Language*
ISEP	Instituto Superior de Engenharia do Porto
ISP	*Interface Segregation Principle*
LSP	*Liskov Substitution Principle*
OCP	*Open-Closed Principle*
PAP	*Policy Administrator Point*
PDP	*Policy Decision Point*
PEP	*Policy Enforcement Point*
PESEE	Projeto/Estágio da Licenciatura em Engenharia Eletrotécnica – Sistemas Elétricos de Energia

PESTA	Projeto/Estágio da Licenciatura em Engenharia Eletrotécnica e de Computadores
PESTM	Projeto/Estágio da Licenciatura em Engenharia Mecânica
PESTQ	Projeto/Estágio da Licenciatura em Engenharia Química
PIP	*Policy Information Point*
PRIN3	Projeto Interdisciplinar III da Licenciatura em Engenharia e Gestão Industrial
PROES	Projeto/Estágio da Licenciatura em Engenharia de Sistemas
PROJI	Projeto Integrado da Licenciatura em Engenharia Civil
QFD	*Quality Function Deployment*
REST	Representational State Transfer
RGPD	Regulamento Geral de Proteção de Dados
RUC	Responsável de Unidade Curricular
RUP	*Rational Unified Process*
SOLID	*Single Responsability Principle, Open-Closed Principle, Liskov Substitution Principle, Interface Segregation Principle, Dependency Inversion Principle*
SRP	*Single Responsability Principle*
SWOT	*Strengths, Weaknesses, Opportunities and Threats*
TMDEI	Tese/Dissertação/Estágio do Mestrado em Engenharia Informática
UC	Unidade Curricular
URL	*Uniform Resource Locator*
XACML	*eXtensible Access Control Markup Language*

1 Introdução

Neste capítulo, serão apresentados:

- o contexto do projeto;
- o problema a que se pretende dar resposta;
- os objetivos que se deseja cumprir;
- a abordagem que foi adotada na resolução do problema.

1.1 Contexto

O Instituto Superior de Engenharia do Porto (ISEP) tem como missão "transmitir conhecimento aos estudantes, ajudá-los a serem confiantes, empreendedores e a competirem com os melhores" (ISEP, 2018a).

Em Portugal, para terminar um mestrado é necessária a realização de uma tese/dissertação/estágio, pelo que todos os cursos de mestrado do ISEP têm uma unidade curricular (UC) deste tipo. Para além dos mestrados, das doze licenciaturas atuais do ISEP, onze têm nos seus planos curriculares uma UC de projeto/estágio.

Em comum, este tipo de UC tem a particularidade do ensino-aprendizagem seguir uma abordagem tutorial, muitas vezes em ambiente organizacional-empresarial, em que o estudante realiza um projeto ou tarefas dirigidas pelo orientador académico e pelo supervisor organizacional. Desta forma, a realização de projetos-estágios curriculares, permite que os estudantes apliquem os conhecimentos adquiridos ao longo dos seus cursos, no contexto organizacional-empresarial em que serão incluídos depois do término destes. A realização destes projetos pressupõe uma forte carga de trabalho cognitivo autónomo por parte do

estudante, mais a aprendizagem de competências muito importantes e de difícil aquisição em ambiente académico.

Ao mesmo tempo, também pressupõe a gestão de todos os processos por parte dos responsáveis das unidades curriculares (RUC) de projeto-estágio, que envolve muitos outros docentes, entidades externas (empresas e outras organizações) e todos os estudantes que as frequentam, o que se traduz numa grande quantidade de trabalho por parte dos RUC. De facto, a maioria dos docentes inquiridos não concorda com o tempo de serviço docente que lhes é atribuído para utilizar nesta gestão (Figura 1).

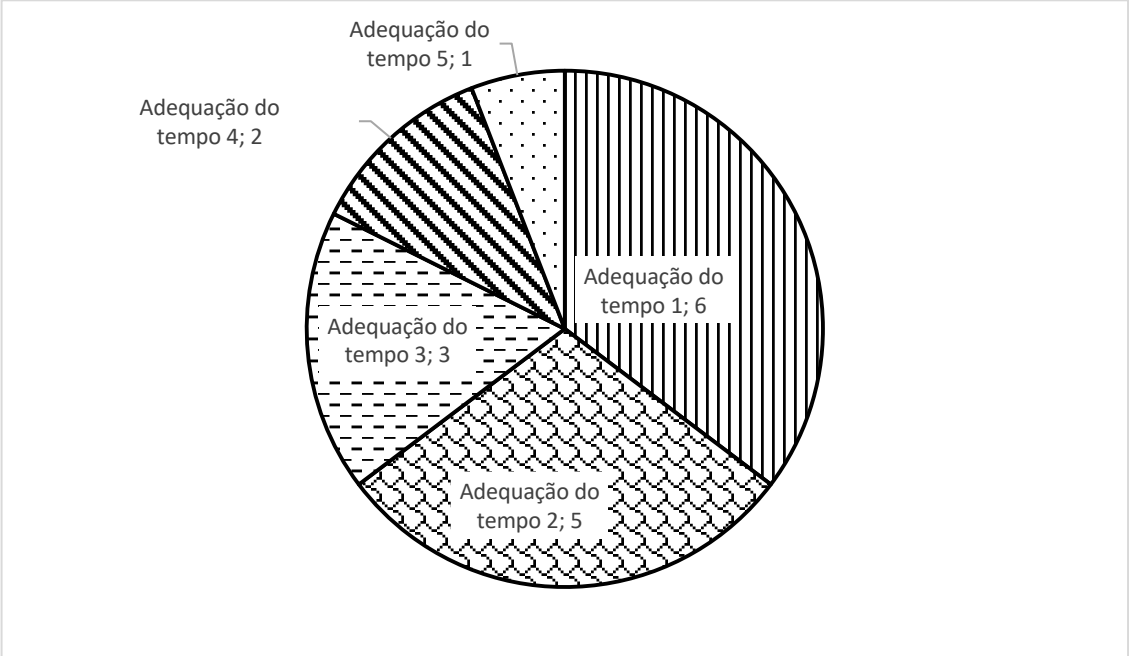

Figura 1 – Adequação do tempo de serviço docente atribuído à gestão da UC

Assim, o facto de o ISEP poder fornecer aos seus docentes um software que os auxilie neste processo, pode significar que conseguem dedicar mais tempo a acompanhar os estudantes e menos nos processos administrativos.

Cada uma destas UC tem o seu próprio funcionamento e as suas etapas específicas, desenvolvidas, adaptadas e aplicadas ao longo do decurso da mesma por cada um dos cursos, não existindo o desejo por parte dos RUC e diretores de curso de uniformizar os diferentes processos.

1.2 Problema

Considerando estas particularidades de cada curso, não existe software de apoio à gestão de muitos dos processos destas UC, e o software que existe (nomeadamente Portal (ISEP, 2019a)

e *Moodle* (ISEP, 2019b)) não está integrado entre si, o que estende e dificulta de sobremaneira as tarefas dos RUC.

Os docentes inquiridos indicam que não existe software de auxílio que integre todas as tarefas de gestão dos projetos (Figura 2).

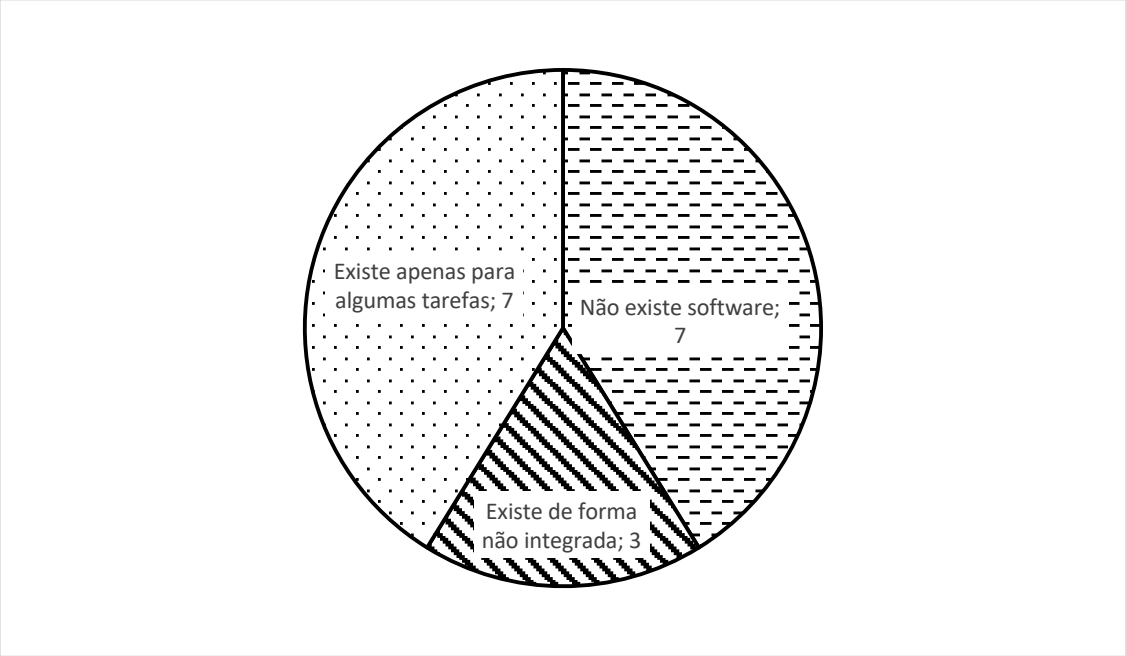

Figura 2 – Conclusões sobre a existência de software de auxílio ("Gestão de projetos curriculares," 2019)

O desenvolvimento de um software que permita a gestão dos projetos curriculares pode traduzir-se numa grande vantagem para o ISEP, pois os docentes que responderam a este questionário concordam com a necessidade de um *software* que permita integrar as tarefas de gestão de projetos curriculares (Figura 3).

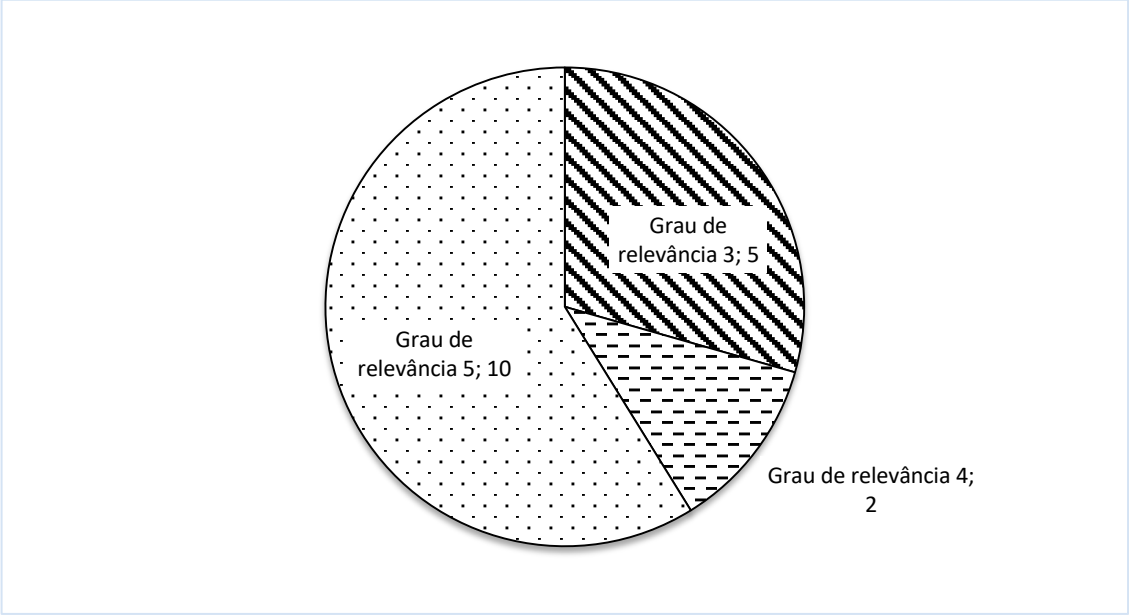

Figura 3 – Relevância do desenvolvimento de um software integrado ("Gestão de projetos curriculares," 2019)

Muitos destes processos continuam a ser geridos em folhas de cálculo (vulgo Excel) com poucos ou nenhuns automatismos de gestão de informação e documentação, rastreabilidade de informação ao longo do processo e comunicação entre intervenientes (e.g. envio de emails).

Para além da quantidade de trabalho manual e cognitivamente muito exigente, os processos e ferramentas usadas não promovem o cumprimento de regulamentos internos (e.g. regulamento de avaliação), externos (e.g. Regulamento Geral sobre a Proteção de Dados (RGPD)) e demais legislação aplicável (e.g. Decreto-Lei nº 65/2018).

1.3 Objetivo

Pretende-se desenvolver um sistema de software que responda às necessidades de gestão de projetos/teses dos cursos do ISEP, e conduza e garanta o cumprimento de leis e regulamentos aplicáveis, suportando os diferentes processos adotados nas diferentes UC.

Este software tem como objetivo manter o funcionamento atual das UC, mas permitir que cada uma delas tenha apoio à gestão dos seus projetos. Se assim o desejarem, os RUC das UC de projeto do ISEP, devem configurar o sistema para que este se adapte ao seu processo de negócio. Esta configuração pode ser alterada a qualquer momento que a UC também altere os seus processos.

A confiabilidade e manutenibilidade do software é de fundamental importância uma vez que, o sistema deve permitir que o RUC consiga realizar os seus processos, independentemente das condições que tem disponíveis (e.g. rede, *hardware*). Também a existência de alterações do

processo de negócio (e.g. alteração da legislação em vigor) ou uma possível melhoria na qualidade do software devem ser precavidas até certo ponto, através da identificação de pontos de evolução e respetivo suporte.

1.4 Abordagem

A abordagem adotada para a resolução do problema apresentado neste documento, é focada no desenvolvimento de um software configurável. Este software deve permitir que as UC de projeto-estágio continuem a funcionar da mesma forma, podendo cada uma manter as suas particularidades e processos atuais. Apesar de especificidades, deverão ser cumpridos regulamentos e legislação aplicável, de forma a que a informação presente nesta aplicação esteja em conformidade com estas regras.

Dado que não existe qualquer abordagem concetual predefinida e muito menos qualquer arquitetura, estas serão estudadas, analisadas, propostas, debatidas e selecionadas ao longo do desenvolvimento do projeto. Como resultado do processo de análise e design, será possível apresentar e selecionar a abordagem concetual e arquitetura que melhor cumpram os objetivos entretanto consolidados.

Para tal, serão desenvolvidos diversos protótipos, que deverão ser apresentados e discutidos com os diversos RUC das UC de projeto-estágio, com vista à validação das decisões tomadas ao longo do processo de desenvolvimento do projeto.

1.5 Contribuições

As principais contribuições deste projeto são as seguintes:

- Sistematização de conceitos, atores e processos de negócio;
- Definição duma arquitetura de sistema de suporte ao negócio, configurável às diferentes formas de funcionamento das UC de projeto-estágio;
- Aplicação servidora (*backend*), que implementa a arquitetura proposta, e disponibiliza funcionalidades capazes de suportar dinamicamente o negócio de várias UC;
- Aplicação *frontend* que permite comprovar a validade da abordagem arquitetural e a sua implementação, ao disponibilizar um subconjunto de funcionalidades de suporte à gestão de processos de negócio de diversas UC configuradas heterogeneamente.

1.6 Estrutura do documento

No primeiro capítulo é descrito o contexto em que o projeto se insere, bem como o problema, os objetivos e a abordagem a adotar.

No segundo capítulo do presente documento, é sistematizado o contexto em maior detalhe, apresentando a legislação aplicável e as soluções já existentes.

No terceiro capítulo é apresentada a engenharia de requisitos, sendo que é neste capítulo que são descritos os métodos de elicitação de requisitos utilizados, são identificadas as partes interessadas e também são sistematizados os requisitos funcionais e não funcionais. Para além disso, neste capítulo é realizada a priorização dos requisitos do sistema.

No quarto capítulo são apresentados os conceitos e processos relacionados com a gestão das UC de projeto-estágio.

No quinto capítulo é apresentado o design do sistema, desde a granularidade mais grossa até à mais fina.

No sexto capítulo são apresentados diversos conceitos presentes na implementação do sistema.

No sétimo capítulo são apresentadas as experiências e avaliação realizadas, bem como a interpretação dos resultados obtidos.

No oitavo e último capítulo são apresentadas as conclusões do projeto desenvolvido.

2 Contexto

Este capítulo tem como objetivo apresentar:

- o contexto de negócio, o que inclui a análise da legislação aplicável à avaliação de projetos curriculares no âmbito de Licenciaturas e Mestrados, como também a legislação referente ao Regulamento Geral de Proteção de Dados (RGPD);
- apresentação dos requisitos não funcionais que o sistema a desenvolver deve cumprir, com vista ao cumprimento dos objetivos e resolução do problema enunciado;
- descrição do contexto tecnológico, identificando as aplicações já existentes e que em algumas UC já são utilizadas no processo de gestão, bem como abordagens tecnológicas que podem ser adotadas na construção do sistema desejado.

2.1 Legislação aplicável

As subsecções seguintes apresentam a legislação aplicável ao problema enunciado, sobretudo relacionadas com a avaliação e atribuição de graus no ensino superior e com o RGPD.

2.1.1 **Atribuição de graus e diplomas do ensino superior politécnico**

Segundo o Decreto-Lei nº 65/2018 (Diário da República Eletrónico, 2018a), um ciclo de estudos que permite obter o grau de licenciado, no ensino politécnico, é constituído por 180 créditos e normalmente tem uma duração de seis semestres curriculares. Este tipo de cursos deve ter como objetivo: fornecer aos alunos todas as ferramentas e conhecimentos necessários para a execução das atividades relacionadas com a área de estudos (Artigo 8º).

Por outro lado, os ciclos de estudos que conferem o grau de mestre são constituídos por 90 a 120 créditos, com a duração de três a quatro semestres curriculares. Ao nível do ensino politécnico, o grau de mestre deve ser dado aos alunos que demonstrem a aquisição de

conhecimentos relacionados com uma determinada especialização, mas também alguma atividade prática no âmbito da investigação (Artigo 18º).

Este decreto-lei também define algumas obrigações que devem ser seguidas na avaliação das dissertações de atribuição do grau de mestre:

- Os estudantes devem ser orientados, no âmbito do desenvolvimento dos projetos finais, por um docente com grau de doutor ou por especialistas na área de realização do projeto;

- A prova final deve ser apreciada e discutida por um júri, nomeado pela entidade legal e ser constituído por três a cinco membros, sendo que um destes pode ser o orientador do estudante;

- Os membros do júri devem ter alguma especialização na área em que se insere o projeto final;

- Como resultado do processo de avaliação final do projeto desenvolvido pelo estudante, deve ser elaborada uma ata, onde sejam identificados os votos de cada membro do júri, bem como as suas respetivas fundamentações.

2.1.2 **Cumprimento do RGPD**

O RGPD é um diploma elaborado pela União Europeia, que visa a proteção, tratamento e circulação dos diversos dados pessoais (e.g. nome, morada, número do cartão de identificação) (Comissão Europeia, 2016). Este diploma pressupõe que todas as empresas ou aplicações produzidas num dos estados membros da União Europeia, protejam os dados dos seus clientes e apenas utilizem estes dados com o consentimento destes (IAPMEI, 2018).

A utilização dos dados pessoais dos utilizadores deve ser consentida por estes, pelo que as empresas e aplicações devem solicitar este consentimento. Uma vez que, o projeto a ser desenvolvido, utilizará dados pessoais tanto de alunos envolvidos nas UC de projeto-estágio, como de empresas que acolhem estes alunos, é necessário que permitam que os seus dados sejam utilizados pelo sistema a ser desenvolvido. Desta forma, existe a necessidade de utilizar tecnologias que permitam que a base de dados utilizada no sistema seja cifrada e que os dados que circulam na aplicação, o façam de uma forma segura, para que não exista nenhum tipo de roubo ou alteração destes dados.

De acordo, com a Resolução do Conselho de Ministros nº 41/2018 (Diário da República Eletrónico, 2018b), existem diversos requisitos técnicos relacionadas com a segurança da informação que circula nos sistemas de informação (Tabela 1).

Tabela 1 – Requisitos técnicos dos sistemas de informação (Diário da República Eletrónico, 2018b)

Requisito geral	**Requisito Específico**		**Classificação**
As aplicações cliente devem ser desenvolvidas adotando práticas de desenvolvimento seguro.	Front-end	Seguir as boas práticas de desenvolvimento.	Obrigatório
		Utilização de sessões seguras com protocolo de segurança	Obrigatório
		Não guardar informação pessoal no *browser*, memória ou disco, para além do tempo de sessão e apenas na medida do necessário	Obrigatório
	Camada aplicacional	Utilização de sessões seguras com protocolo de segurança.	Obrigatório
		Se possível usar certificados através de *Application Programming Interface* (API), não sendo desta forma necessário o uso de palavras-passe.	Recomendado
Capacidade de autenticar e autorizar todos os utilizadores e dispositivos, incluindo o controlo de acesso a sistemas e aplicações.	Front-end	O processo de autenticação deve ser sempre iniciado e mantido em sessão segura.	Obrigatório
		Dados pessoais de sessão excluídos das variáveis *Uniform Resource Locator* (URL) ou de outras variáveis visíveis ao utilizador.	Obrigatório
		Recomenda-se que para novos sistemas seja sempre usado como padrão de autenticação o 2FA.	Recomendado
Atribuição de direitos de acesso e privilégio de forma restrita e controlada.	*Front-end,* Camada Aplicacional e Base de dados	Criação de perfis com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal, de acordo com o princípio da necessidade de conhecer	Obrigatório
		Criação de registo de acesso, alteração e remoção (*logs*), com informação sobre quem acedeu, de onde acedeu (IP e Porto), quando acedeu, a que dados acedeu, que ação foi	Obrigatório

		efetuada sobre os mesmos (CRUD).	
Restrição de acesso à informação baseado no princípio da necessidade de conhecer (criação de perfil).			Obrigatório
Automatização dos processos de concessão, revisão, análise e revogação de acesso.			Obrigatório
Procedimentos seguros de início de sessão.			Obrigatório
Capacidade de monitorização, registo e análise de toda a atividade de acessos de modo a procurar ameaças prováveis.	Deve ser guardado registo de atividade (log) de todas as ações que um utilizador efetue sobre dados pessoais, independentemente do seu perfil e função.		Obrigatório
	Todos os registos de atividade (log) devem ser armazenados apenas em modo de leitura, devendo, com uma periodicidade máxima de 1 mês, ser englobados num único bloco de registos e assinado digitalmente (garantia de integridade).		Obrigatório
	Deve ser guardado registo de atividade (log) de todos os acessos e tentativas falhadas de acesso, obedecendo aos requisitos anteriores.		Obrigatório
	Garantir que os registos de atividade provenientes dos diversos subsistemas (Sistemas Operativos, aplicações, browsers, Sistema de Gestão de Base de Dados — SGBD, etc.) são inequivocamente associados à sua origem.		Obrigatório
	Os registos de atividade (log) devem conter, no mínimo, o endereço de acesso (IP e Porto), Host, HASH da conta do utilizador que efetuou a ação, ação efetuada (CRUD), Tipo de Dado Pessoal onde a ação foi efetuada, data/hora/minuto/segundo (TimeStamp) da ação, alteração efetuada sobre o dado pessoal.		Obrigatório
Inspeção automática dos conteúdos para procurar dados sensíveis e acessos remotos ao sistema a partir do exterior do ambiente organizacional.			Obrigatório
Proteção dos dados contra modificações não autorizadas, perdas, furtos e divulgação não autorizada.	Front-end	FE desenvolvido e em produção de acordo com as melhores práticas de segurança, garantindo a proteção desta camada aos ataques mais comuns (SQLi, injeção de código, etc.).	

	Camada aplicacional	Camada aplicacional segregada da rede ou ambiente com visibilidade e/ou acesso exterior.	Obrigatório
Os sistemas de armazenamento devem garantir redundância e disponibilidade, não devendo existir nenhum «*single point of failure*».			Obrigatório

2.2 Soluções existentes

Nas próximas secções são apresentadas as aplicações que já existem no processo de gestão de projetos, mas que não funcionam de forma integrada entre si.

2.2.1 **Moodle ISEP** (ISEP, 2019b)

O *Moodle* ISEP é a plataforma utilizada tanto por estudantes como por docentes no decurso das UC. Este permite que todo o conteúdo que os docentes pretendem que chegue aos estudantes, seja partilhado com estes. Existe a possibilidade de os estudantes entrarem em contacto com os seus pares, mas também com os diversos docentes que lecionam a UC. O *Moodle* ISEP contem permissões relacionadas com o papel de estudante e de docente (RUC ou não), o que permite que um dado trabalho/documento que seja inserido nesta plataforma só seja consultado pelos demais se as permissões assim o possibilitarem.

2.2.2 **Portal ISEP** (ISEP, 2019a)

O Portal ISEP (ISEP, 2019a) possui toda a informação administrativa relativa ao estudante, mas também relacionada com o funcionamento do ISEP. É a plataforma onde os alunos devem submeter a entrega final do projeto, para que futuramente este documento seja validado pelo orientador e homologado pela Presidência do ISEP e enviado para o depósito de teses do IPP, sendo o último apenas aplicado a teses de mestrado.

2.2.3 **GestProj/GestMEI** (DEI, 2019a) (DEI, 2019b)

Estas plataformas permitem que entidades externas ao ISEP ou até mesmo docentes do ISEP, submetam propostas de projetos a realizar no âmbito das UC mencionadas. Estas propostas incluem a duração da realização do projeto, a descrição e parte da abordagem a utilizar. Desta forma, os estudantes podem consultar as propostas que mais lhes interessam e proceder ao contacto com o proponente. As propostas presentes nestas plataformas podem ser ou não revistas pelo RUC, com vista à validação da dimensão do problema e da adequação dos objetivos e abordagem ao âmbito do curso em que se inserem os projetos.

2.2.4 **PRAXIS** (PRAXIS, 2019)

O *European Centre for Project/Internship Excellence* (PRAXIS) é uma plataforma que permite que os alunos pesquisem estágios e que desta forma possam ser integrados no mercado de trabalho. Possibilitando a submissão de propostas de projeto para diferentes níveis de ensino, o PRAXIS permite aos estudantes e às empresas terem um contacto mais direto e personalizado, uma vez que, esta plataforma possui diversos serviços, tanto para permitir que o perfil do estudante seja destacado, como também que as empresas encontrem o candidato que procuram, entre outros (ISEP, 2018b).

2.2.5 **Aplicação *NodeJS***

A aplicação *NodeJS* foi desenvolvida à medida das necessidades encontradas pelo RUC da UC de Tese/Dissertação/Estágio do Departamento de Engenharia Informática (TMDEI). Esta aplicação não possui preocupações com confidencialidade da informação nem permite o acesso dos estudantes à informação referente à própria tese. Permite que o RUC organize os projetos, nomeadamente nas diferentes entregas envolvidas nesta UC e comunique as revisões destes aos respetivos orientadores. Não possui ligação a qualquer outra aplicação, pelo que toda a informação tem de ser introduzida manualmente ou através de ficheiros XSLX, com recurso a informações presentes (nomeadamente) no *Moodle* e Portal do ISEP.

2.2.6 **Interligação de sistemas**

A Figura 4 representa as relações existentes entre as diversas aplicações, anteriormente mencionadas. Apesar de existirem relações entre o *Moodle* ISEP, Portal ISEP e Aplicação *NodeJS*, esta comunicação é realizada através de partilha de ficheiros XSLX resultantes da exportação da informação destas aplicações.

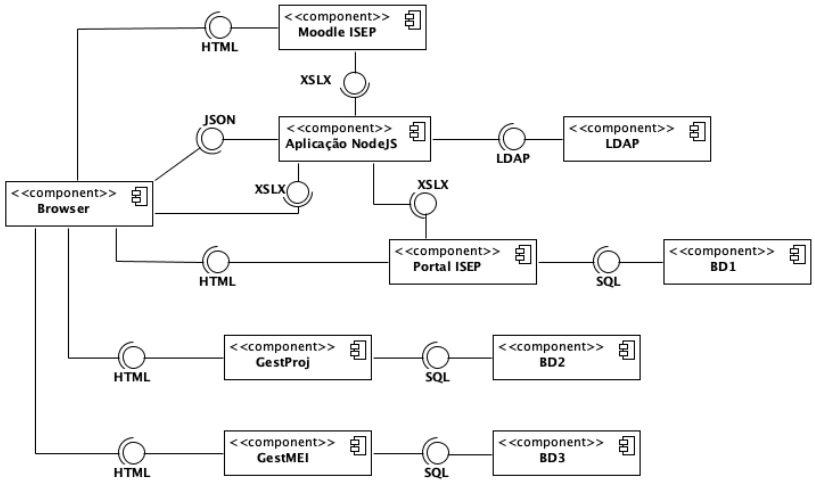

Figura 4 – Representação arquitetural dos sistemas existentes de suporte ao negócio

2.3 Síntese

Este capítulo descreveu sumariamente a legislação aplicável e o contexto tecnológico existente que inspira este projeto, tanto pelas funcionalidades de que dispõe como pelas limitações que apresenta.

3 Engenharia de requisitos

Este capítulo descreve as seguintes atividades:

- Elicitação de requisitos através do inquérito a RUC e diretores de cursos;
- Identificação e caraterização das partes interessadas nos processos elicitados;
- Sistematização de requisitos funcionais;
- Sistematização de requisitos não funcionais.

3.1 Elicitação de requisitos

A elicitação de requisitos deste projeto foi realizada com recurso a:

- Inquérito aos RUC das UC de projeto-estágio e dos diretores de curso;
- Realização de entrevistas aos mesmos RUC, de forma a complementar a informação recolhida com o questionário.

3.1.1 **Inquérito**

Para que os processos das diferentes UC fossem conhecidos e posteriormente analisados, foi elaborado um inquérito (Anexo A – Inquérito sobre Gestão de Projetos Curriculares). Este inquérito foi enviado via email aos RUC das UC de projeto-estágio e aos diretores de curso do ISEP.

Inicialmente, foi necessário verificar se existia de alguma forma a possibilidade das UC terem processos únicos, isto é, todas as UC terem o mesmo número de entregas parciais ou o processo de escolha e formalização das propostas ter o mesmo funcionamento, por exemplo. Esta

possibilidade não teve grande aceitação, dado que a maioria das UC já funciona da mesma forma há algum tempo e esta nova abordagem obrigaria à alteração dos processos já conhecidos pelos docentes e até mesmo pelos estudantes, uma vez que alguns frequentam estas UC mais que uma vez. Portanto, não existe ambição de unificar os processos de gestão dos projetos.

3.1.2 **Reuniões**

De forma a complementar a informação recolhida através do inquérito, foram realizadas entrevistas com alguns RUC das UC de projeto e diretores de curso:

- Projeto/Estágio da Licenciatura em Engenharia Mecânica (PESTM);
- Projeto/Estágio da Licenciatura em Engenharia Química (PESTQ) e Estágio da Licenciatura em Biorrecursos (ESTBI);
- Projeto/Estágio da Licenciatura em Engenharia de Sistemas (PROES);
- Dissertação/Estágio do Mestrado em Engenharia Química (DISEST);
- Projeto Integrado da Licenciatura em Engenharia Civil (PROJI) & Dissertação/Projeto/Estágio do Mestrado em Engenharia Civil (DIPRE);
- Projeto/Estágio da Licenciatura em Engenharia Eletrotécnica – Sistemas Elétricos de Energia (PESEE);
- Projeto Interdisciplinar III da Licenciatura em Engenharia e Gestão Industrial (PRIN3);
- Projeto/Estágio da Licenciatura em Engenharia Eletrotécnica e de Computadores (PESTA).

Estas reuniões tiveram como principais objetivos:

- Elicitar e analisar processos de gestão das UC de projeto-estágio;
- Expor a visão do projeto e receber sugestão de valor, prioridades e riscos.

Como resultado, foram elaboradas atas (Anexo B – Atas das reuniões realizadas com os RUC), que permitiram registar as opiniões e sugestões dos docentes.

A realização do inquérito e das reuniões permite conhecer as diferentes abordagens encontradas em cada uma das UC envolvidas. Desta forma, é possível conhecer a que ações os RUC atribuem mais importância e que pretendem ver refletidas no software final.

Por isso, as secções seguintes focam-se na abordagem às partes interessadas do projeto e aos requisitos que foi possível sistematizar da realização destes métodos de elicitação de requisitos.

3.2 Partes interessadas

Durante a análise dos questionários e do conteúdo das reuniões foram identificadas diversas partes interessadas nos processos de gestão das UC de projeto, sistematizadas na lista seguinte:

- ISEP – instituição de ensino responsável pela graduação, para tal proporcionando aos estudantes competências e conhecimentos aplicados aos seus cursos e assim gerar engenheiros aptos ao mundo empresarial. Tem interesse em facilitar os processos de gestão letivos, nomeadamente através de software;

- Departamento – parte estrutural e organizacional do ISEP, ao qual estão afetos cursos, docentes e instalações, onde se realizam muitos dos projetos e provas avaliações;

- Curso – conjunto de Unidades Curriculares (UC) organizadas num plano curricular em que o estudante está inscrito, e que tem uma UC de projeto;

- Discente – pessoa que se encontra a frequentar um curso e que irá realizar um projeto, e que é também ator do sistema;

- Docente – entidade que leciona a UC de projeto, pode desempenhar outras funções, nomeadamente orientador, coorientador, revisor, proponente, arguente e presidente de prova;

- Diretor de curso – docente responsável por gerir o curso, acumulando, por vezes, esta função com a de RUC (cf. abaixo) da UC de projeto do curso, e de presidente de júri (cf. abaixo) de muitas (por vezes todas as) provas de avaliação;

- Responsável por UC (RUC) – docente responsável por uma determinada unidade curricular (UC), nomeadamente da UC de projeto. O RUC pode, nalguns casos, desempenhar outros papéis complementares, nomeadamente o de orientador, arguente ou presidente de prova;

- Diretor de departamento – docente responsável pela gestão do departamento;

- Organização externa – empresa que proporciona aos estudantes a realização de projetos num ambiente não académico. Estas organizações são muitas vezes contactadas pelos diversos docentes envolvidos nas UC, mas noutros casos são as próprias organizações que entram em contacto com o ISEP para proporem projetos.

- Orientador – docente responsável por auxiliar o estudante no processo de desenvolvimento do seu projeto;

- Coorientador – docente, que em cooperação com o orientador, auxilia o estudante no processo de desenvolvimento do seu projeto, não existindo em todos os projetos;

- Supervisor – funcionário da entidade externa onde o projeto se realiza que supervisiona o trabalho do estudante em ambiente empresarial. Alguns projetos não são realizados em estágio, pelo que não têm supervisor;

- Proponente – funcionário de uma organização externa ou um docente do ISEP que submete uma proposta de projeto;

- Arguente – docente do ensino superior ou investigador que avalia o projeto realizado, nomeadamente, discutindo com o estudante as decisões tomadas ao longo do desenvolvimento do projeto;

- Revisor – docente responsável por verificar as entregas dos estudantes e retificar se os aspetos desenvolvidos se encontram de acordo com os objetivos do curso;

- Presidente de prova – docente responsável por dirigir os trabalhos da prova de avaliação, o que pode incluir a elaboração da ata de prova. Muitas das vezes, o presidente de prova é também o RUC da UC;

- Atribuidor – pessoa responsável (RUC ou proponente) por atribuir um discente a uma proposta ou projeto;

- Estudante – discente que irá realizar uma proposta ou projeto.

A Tabela 2 representa a sistematização dos atores do sistema, indicando se as partes interessadas (secção anterior) são ou não atores do sistema.

Tabela 2 – Sistematização dos atores do sistema

Parte Interessada	**Ator**
ISEP	Não
Departamento	Não
Curso	Não
Discente	Sim
Docente	Sim
Diretor de curso	Não
Responsável por UC (RUC)	Sim
Diretor de departamento	Não
Organização externa	Não
Orientador	Sim
Coorientador	Sim
Supervisor	Sim
Proponente	Sim
Arguente	Sim
Revisor	Sim

Presidente de prova	Sim
Atribuidor	Sim
Estudante	Sim

3.3 Requisitos funcionais

Os requisitos funcionais estão intimamente relacionados com funcionalidades com interação dos utilizadores do *software*.

Das respostas ao questionário e das reuniões com os RUC, identificam-se seis grupos de casos de uso:

- Gestão de proposta;
- Seleção, Atribuição e Formalização;
- Gestão de projeto;
- Entregas;
- Marcação de prova;
- Avaliação.

A Figura 5 apresenta o diagrama de casos de uso elicitados, agrupados segundo a categorização anterior.

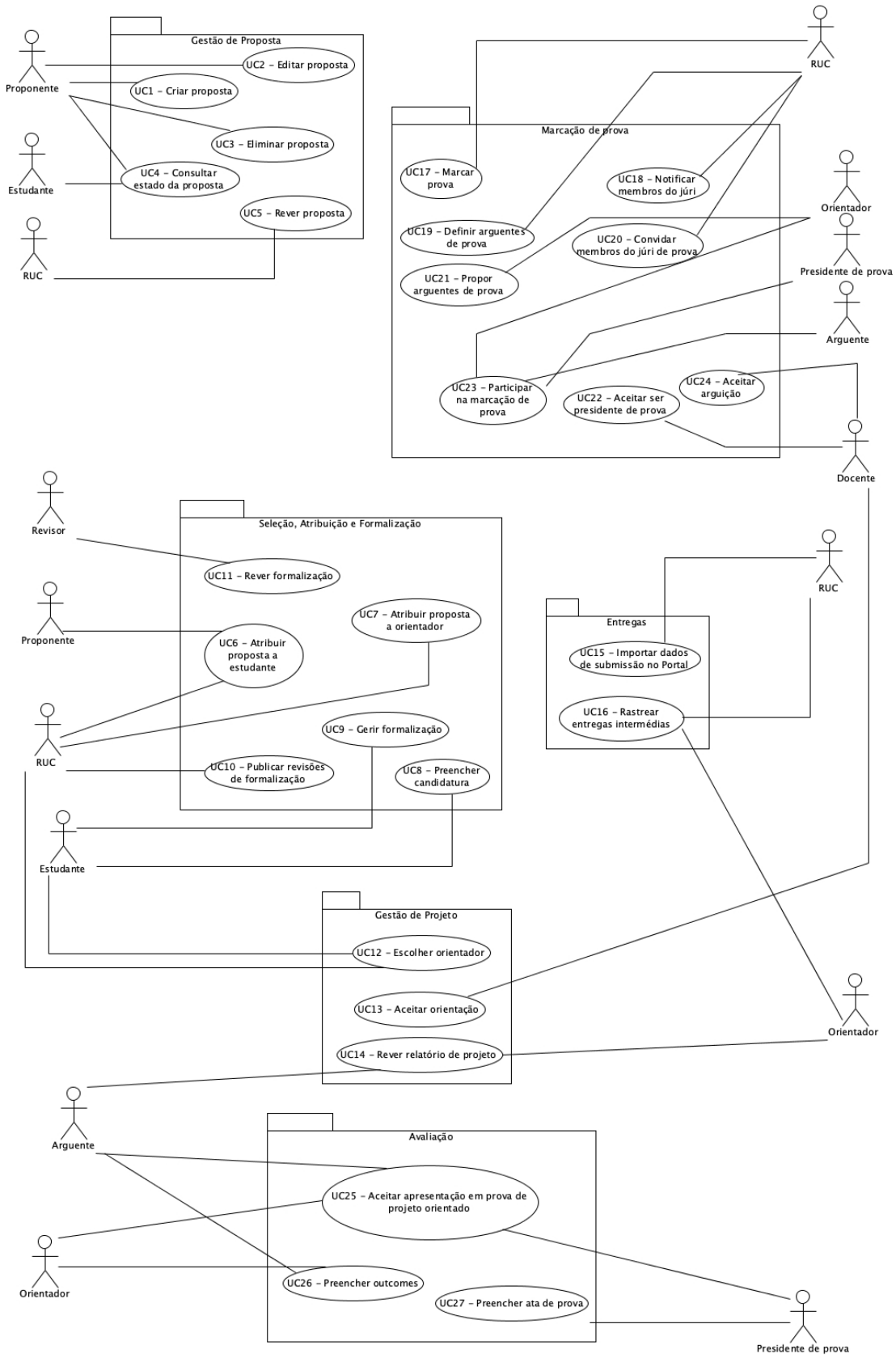

Figura 5 – Diagrama de casos de uso

Também a partir de informação recolhida dos questionários e inquéritos, sistematizaram-se os casos de uso por UC (Tabela 3). As descrições dos casos de uso encontram-se no Anexo C – Descrição dos requisitos funcionais.

Tabela 3 – Sistematização dos requisitos funcionais

Requisito Funcional	**PESTI**	**TMDEI**	**PESTM**	**PESTQ & ESTBI**	**PROES**	**DISEST**	**PROJI & DIPRE**	**PESEE**	**PRIN3**	**PESTA**	**total**
UC1	X	X	X	X	X	X	X	X	X	X	9
UC2	X	X	X	X	X	X	X	X		X	8
UC3	X	X	X	X	X	X	X	X		X	8
UC4			X		X	X	X	X			5
UC5	X	X	X	X	X	X	X	X	X	X	9
UC6	X	X	X	X	X	X	X	X	X	X	9
UC7	X	X			X		X	X	X	X	7
UC8			X		X	X	X		X	X	6
UC9		X									1
UC10		X									1
UC11		X									1
UC12	X	X	X	X	X	X	X	X	X	X	9
UC13	X	X	X	X	X	X	X	X	X	X	9
UC14	X	X	X	X	X	X	X	X	X	X	9
UC15	X	X		X	X		X	X	X		7
UC16		X		X	X	X			X		5
UC17	X	X	X	X	X	X	X	X	X	X	9
UC18	X	X	X		X	X	X		X	X	7
UC19	X	X	X	X	X	X	X	X	X	X	9
UC20		X	X	X		X	X				5
UC21		X					X		X		3
UC22	X	X			X	X	X	X	X	X	8
UC23	X	X	X	X	X	X	X	X	X		9
UC24	X	X	X	X	X	X	X	X	X		9
UC25							X				1
UC26	X	X	X	X		X					5
UC27	X	X				X					3
total	18	24	17	16	19	20	21	16	17	14	

A Tabela 4 representa a sistematização dos grupos de casos de uso que estão presentes em cada UC estudada.

Tabela 4 – Sistematização dos grupos de casos de uso

Grupo de casos de uso	**PESTI**	**TMDEI**	**PESTM**	**PESTQ & ESTBI**	**PROES**	**DISEST**	**PROJI & DIPRE**	**PESEE**	**PRIN3**	**PESTA**	**total**
Gestão de proposta	X	X	X	X	X	X	X	X	X	X	10
Seleção, Atribuição e Formalização	X	X	X	X	X	X	X	X	X	X	10
Gestão de projeto	X	X	X	X	X	X	X	X	X	X	10
Entregas	X	X	X	X	X	X	X	X	X	X	10
Marcação de prova	X	X	X	X	X	X	X	X	X	X	10
Avaliação	X	X	X	X	X	X	X				7
total	6	6	6	6	6	6	6	5	5	5	

3.4 Requisitos não-funcionais

Se no desenvolvimento de software existem diversos requisitos funcionais que são definidos ou desejados pelo cliente para uso pelos utilizadores (cf. secção anterior), por outro lado, existem requisitos que não estão intimamente ligados às funcionalidades a disponibilizar ao utilizador, mas à forma do sistema de software e à forma como é desenvolvido e mantido.

É necessário que estes requisitos sejam classificados/categorizados com vista à sua sistematização e posterior análise e aplicação no desenvolvimento do software. Para tal, será adotado o modelo FURPS+ (Eeles, 2005) que sugere as seguintes categorias: (i) funcionalidade, (ii) usabilidade, (iii) confiabilidade, (iv) desempenho, (v) suportabilidade, (vi) desenho, (vii) implementação, (viii) implantação e (ix) interligação.

3.4.1 **Funcionalidade**

Contrariamente, aos requisitos funcionais (cf. Secção 3.3), os requisitos não-funcionais na categoria de funcionalidade não são específicos do domínio/negócio, mas tipicamente transversais a diversos projetos/negócios. Nesta categoria são incluídos requisitos, tais como: Autenticação e Autorização, Auditoria, Localização (e Internacionalização) ou outros relacionados com ações do sistema (e.g. envio de emails, impressão, relatórios, etc.) (Eeles, 2005).

Nas secções seguintes são descritos os requisitos de funcionalidade necessários para o desenvolvimento do sistema mencionado.

3.4.1.1 Autenticação

Autenticação é o processo de uso de credenciais para que o sistema comprove a identidade do utilizador dessas credenciais (Microsoft, 2018).

Dada a diversidade de utilizadores que terão acesso ao sistema, é necessário garantir que cada um deles é reconhecido pelo mesmo e que apenas acede às informações que o seu perfil permite.

Também é requerido que as credenciais dos utilizadores do ISEP sejam válidas neste sistema. Complementarmente, de forma a aumentar a confiança dos utilizadores na aplicação, é necessário utilizar um sistema de autenticação já existente, e não um sistema desenvolvido ou privado da aplicação. Ou seja, pretende-se que o sistema a desenvolver não seja responsável pela leitura e manipulação de credenciais, mas que usufrua das credenciais e autenticação realizada por outro sistema.

A ser possível, tal funcionalidade traduz-se em:

- redução da sobrecarga cognitiva do utilizador, pois este não tem que saber/usar mais credenciais;

- maximiza a confiança no uso desta aplicação, pois as credenciais não são manipuladas pela aplicação, mas por um sistema no qual já tenha confiança.

3.4.1.2 Autorização

A autorização é o processo que garante que só utilizadores com as devidas permissões acedem a informação ou realizam tarefas (IBM, 2014a).

A execução de praticamente todos os casos de uso acima mencionados apenas é permitida aos utilizadores registados no sistema, mas há casos de uso que só podem ser realizados por um subconjunto de atores. E.g. em determinadas UC, a submissão de propostas é realizável apenas por externos ou docentes.

Além disso, porque o sistema deve ser capaz de lidar com diferentes UC com modos de organização e funções específicas, as autorizações devem ser configuráveis por UC e edição. E.g. a submissão de formalização (processo/documento existente apenas em algumas UC) é autorizada apenas a estudantes.

Acresce que, as autorizações de realização de certas funcionalidades não estão dependentes apenas da identidade e papéis do utilizador (e.g. estudante, docente, externo), mas do contexto de realização da funcionalidade. E.g. o acesso ao documento em avaliação está acessível a docentes e externos, mas só se fizerem parte do júri da respetiva avaliação/prova.

Em síntese, a autorização está dependente dos papéis estáticos dos utilizadores (e.g. estudante, docente, externo) mas também de papéis dinâmicos definidos em função do contexto (e.g. elementos de júri na prova).

3.4.1.3 Auditoria e Rastreabilidade

Auditoria é o processo que analisa o cumprimento de processos (estabelecidos/formais), através da recolha de informação/factos capturados (ou ausentes) durante os processos (IBM, 2019).

Para além disso, a rastreabilidade pode ser utilizada para acompanhar as ações que são realizadas no sistema ou para controlar de que forma é que estas são espoletadas. O uso da rastreabilidade pode resultar na elaboração de relatórios que permitam analisar quando, como e quem realizou estas ações (IBM, 2014b).

Uma das assunções do projeto é a promoção da adoção e cumprimento do RGPD, que obriga o sistema a dispor de mecanismos de auditoria, o que, por sua vez, implica a capacidade de rastreabilidade.

3.4.2 **Usabilidade**

A usabilidade pode ser apresentada como a facilidade que um utilizador tem em realizar ações num determinado sistema.

Este requisito não foi estudado em detalhe, mas, das reuniões com as diversas partes interessadas contactadas (essencialmente RUC), infere-se que:

- O software será usado maioritariamente em écrans superiores a 11 polegadas;
- Tarefas repetitivas devem ser minimizadas;
- Deve haver autoajudas imediatas (e.g. dicas em texto acerca dos ícones de ações);
- Suporte para múltiplas línguas, nomeadamente Português e Inglês.

Embora as sugestões ou recomendações tivessem sido vagas, as observações efetuadas sugerem que esta dimensão do software pode ser fundamental na decisão da sua adoção ou abandono.

Este requisito não funcional é muito difícil de medir (Nielsen, 2012) automaticamente, mesmo com métricas bem definidas, pois é, ainda, muito difícil capturar a opinião de cada utilizador em código. Um sistema em que seja complicado encontrar forma de realizar determinada ação ou que não esteja devidamente organizado, pode colocar em causa o seu sucesso ou insucesso (Nielsen, 2012).

Devido à sua natureza subjetiva e âmbito do projeto e à sua definição informal, acordou-se fazer demonstrações e avaliações periódicas com os utilizadores do sistema para aferir a sua usabilidade, e a partir daí conduzir as decisões de desenvolvimento sobre esta dimensão.

3.4.3 **Confiabilidade**

A confiabilidade pode ser entendida como a capacidade de um software operar sem falhas num determinado período. É relevante que um sistema não funcione apenas em condições ideais, mas também em situações críticas, por exemplo baixa disponibilidade de rede ou excesso de carga no sistema.

Neste projeto, os tipos e respetivos graus de confiabilidade desejados não foram formalmente estipulados, com exceção da necessidade de cumprimento das funcionalidades.

3.4.4 **Suportabilidade**

A suportabilidade inclui, para além de outros, conceito como: configurabilidade, adaptabilidade e manutenibilidade.

A configurabilidade e a adaptabilidade não foram requisitos elicitados, mas dadas as características do domínio e assunções iniciais do projeto, são caraterísticas fundamentais do sistema, como se descreve de seguida.

3.4.4.1 Configurabilidade

A configurabilidade é definida como a capacidade que um cliente tem de personalizar ou parametrizar o *software* com pouca ou nenhuma participação da equipa de desenvolvimento (Mandacina, 2016). Isto é, o produto final deve permitir que o cliente altere (limitadamente) o funcionamento do software.

Uma vez que, as UC de projeto-estágio têm funcionamentos diferentes, existe a necessidade de desenvolver um sistema configurável, que desta forma permita dar suporte a estas desigualdades.

3.4.4.2 Adaptabilidade

A adaptabilidade pode ser vista, como a capacidade de um software se adaptar a alterações no meio ambiente em que se insere, isto é, possíveis mudanças no negócio ou até mesmo ser capaz de ignorar determinadas ações do utilizador que não tenham relevância para o negócio (Subramanian and Chung, 2001a).

A adaptabilidade pode ser aplicada em dois momentos distintos (Jameson, 2001):

- Durante o desenvolvimento, dito software adaptável. A arquitetura deve ter em conta a inserção de novos requisitos. Este tipo de adaptabilidade deve ser tipo em consideração nas disciplinas de Análise e Design do software, para que seja possível reduzir os custos de manutenção do mesmo (Subramanian and Chung, 2001b);

- Em tempo de operação, dito software adaptativo. Este tipo de adaptabilidade permite que o futuro do software seja influenciado pelas decisões ou ações tomadas pelo utilizador, ou seja, o comportamento altera-se conforme o contexto (Jameson, 2001).

Após a análise dos resultados decorrentes das reuniões com os RUC das UC de projeto-estágio, é possível concluir que adaptabilidade em tempo de desenvolvimento tem um papel importante, uma vez que alguns mecanismos atuais do processo de gestão podem ser substituídos no futuro por novos. Por outro lado, a adaptabilidade em tempo de operação não foi muito considerada na elicitação, mas é importante que o design esteja atento a pontos de evolução que suporte a adaptação do sistema à diversidade de partes interessadas e seus processos assumidamente distintos e atualmente desconhecidos.

3.4.4.3 Manutenibilidade

A manutenibilidade é apresentada como a facilidade com que um *software* é alterado. Esta pode ser de quatro tipos diferentes (Visser, 2015):

- Identificação e correção de erros (corretiva);
- Alterações do software relacionadas com adaptações ao negócio (adaptativa);
- Alterações dos requisitos funcionais (perfectiva);
- Aumento do nível de qualidade do software e prevenção de erros futuros (preventiva).

Neste projeto, a manutenibilidade é importante em todos os tipos anteriores, pois o software deve permitir auxiliar diferentes processos variáveis e formalmente mal ou nada definidos, pode resultar na necessidade de corrigir ou estender o software ao longo do seu ciclo de vida, de forma que a configurabilidade e adaptabilidade, mencionadas acima, não sejam capazes de resolver.

Assim, se o sistema for desenvolvido com especial atenção à sua manutenibilidade, permitirá que este processo seja mais simples, realizável por desenvolvedores que não os originais, e com menos encargos.

3.4.5 **Desempenho**

O desempenho está relacionado com medidas como: tempo de resposta ou tempo de recuperação (Eeles, 2005).

Foi definido o seguinte objetivo:

- Tempo de resposta em todos os pedidos ser inferior a 1 segundo, em condições realistas de utilização do sistema pelos utilizadores potenciais.

3.4.6 **Requisitos de Design**

Assumem-se três requisitos de design:

- Potenciar a Modularidade do sistema;
- Aplicação multiutilizador;
- Adoção de TDD.

3.4.6.1 Modularidade

Modularidade é a separação dos componentes de um sistema em módulos independentes, de forma a que cada um deles seja responsável única e exclusivamente por uma responsabilidade (Parnas, 1972).

Assume-se a necessidade de desenvolver um sistema de software modular, que, portanto, facilite e potencie a modularidade e adaptabilidade.

Para tal, é necessário adotar padrões e boas práticas fundamentais de design de software, como seja:

- GRASP (Larman, 2005) nomeadamente,
    - Alta Coesão e Baixo Acoplamento, porque permite a separação de responsabilidade, não criando relações por vezes desnecessárias
- SOLID (Martin, 2000)
    - SRP: *Single Responsability Principle*, cada componente/interface deve ser responsável única e exclusivamente pelas suas atividades. A utilização deste princípio permite que as responsabilidades sejam separadas, permitindo assim um desenvolvimento de componentes reutilizáveis;
    - OCP: *Open-Closed Principle*, não deve existir a necessidade de alterar um módulo, para que este se adapte à função que pretendemos;
    - LSP: *Liskov Substitution Principle* principalmente focado na hierarquia de classes. A utilização de uma classe filha deve ser igual à utilização de uma classe mãe;
    - ISP: *Interface Segregation Principle*, representado como a capacidade de uma classe se apresentar aos seus diferentes clientes de forma adaptada a cada um deles;

    - DIP: *Dependency Inversion Principle*, a dependência de abstrações é preferida à utilização de implementações.

- DRY: *Don't Repeat Yourself*, é um princípio, que permite desenvolver software mais confiável e de mais fácil manutenção.

3.4.6.2 Aplicação multiutilizador

Considerando a descentralização de responsabilidades por UC e por atores, assume-se desde já o desejo de desenvolver um software no estilo cliente-servidor e baseado em standards web, e.g. HTTP/S, HTML, CSS, XML, JSON, REST/SOAP (Berners-Lee, 1994).

3.4.6.3 TDD

Também é desejado que seja seguida uma abordagem TDD (Beck, 2000). Este tipo de abordagem prevê o design e escrita dos testes antes da implementação do código base da aplicação, garantindo que todo o código desenvolvido cumpre o teste desenhado, e usufruindo, como consequência, de testes automáticos de regressão, o que potenciem a manutenibilidade.

3.4.7 Requisitos de Implementação

Assumem-se dois requisitos de implementação descritos nas subsecções seguintes.

3.4.7.1 Continuous Integration/Continuous Delivery (CI/CD)

Assume-se a necessidade de utilização de um pipeline CI/CD que permita automatizar os processos de entrega final do software à medida que novas funcionalidades são desenvolvidas ou corrigidas.

Continuous Integration (CI) é uma prática de desenvolvimento de software que visa integrar constantemente/continuamente as diversas alterações ao código fonte (Anastasov, 2019), garantindo a sua constante/contínua correção e prontidão.

Continuous Delivery (CD) complementa CI ao promover a automatização do processo de instalação e configuração do software em diversos ambientes, incluindo no de produção.

3.4.7.2 Tecnologia de construção/implementação

O único requisito de tecnologia de construção/implementação expressamente definido, é que esta seja de uso livre/grátis.

3.4.8 Requisitos de Interligação (com outros sistemas)

Visto que, em algumas UC inqueridas existe a submissão do documento final do projeto no Portal ISEP, é desejado que o sistema a desenvolver se interligue com este. Esta interligação

pode permitir reduzir o processo manual de importação dos documentos e assim ajudar a facilitar o processo de gestão dos projetos.

Para além disto, os utilizadores devem autenticar-se no sistema usando as credenciais de que dispõem nos sistemas do ISEP (e.g. Portal, Moodle). Desta forma, deve existir uma interligação do sistema com um mecanismo de autenticação que permita a autenticação dos utilizadores perante o sistema a desenvolver.

3.4.9 **Requisitos Físicos**

O sistema a desenvolver tem os seguintes requisitos físicos:

- A aplicação servidora deve ser instalada numa máquina (virtual) com acesso público por HTTP/S que permita o acesso às funcionalidades de negócio, e por protocolos e portas que permitam a gestão do servidor, mas inacessível por qualquer outro meio;

- A utilização da interface (aplicação cliente) por parte dos atores far-se-á maioritariamente em écrans de tamanho igual ou superior a 11 polegadas.

3.4.10 **Síntese**

Apesar do processo de elicitação ter sido extenso e demorado, não houve a pretensão de identificar todos os requisitos aplicáveis, mas essencialmente a identificação daqueles que mais relevante para o negócio eram, e que mais impacto arquitetural possam ter.

3.5 Priorização dos requisitos do sistema para o negócio

Os requisitos de um sistema de software necessitam de ser priorizados, para que desta forma seja possível tomar decisões, nomeadamente de design arquitetural. Habitualmente são utilizadas dois fatores de priorização (Corporation, 2001a):

- Benefício para as partes interessadas, i.e. afere a importância que determinado requisito tem para as partes interessadas (Corporation, 2001b);

- Impacto arquitetural de cada requisito.

Por agora, neste contexto, apenas serão aferidos os benefícios para as partes interessadas, sendo o impacto arquitetural de cada requisito analisado durante o processo de design arquitetural (cf. Capítulo 5).

O benefício de cada requisito do sistema para as partes interessadas é classificado como:

- Crítico, quando tal requisito é considerado uma tarefa principal do sistema, e que são muitas vezes utilizados;

- Importante, quando tal requisito permite que o sistema cumpra as suas principais funções, mas menos relevantes que os requisitos críticos;

- Útil, quando tal requisito não é relativo aos principais objetivos do sistema, mas auxiliam na realização dos restantes requisitos funcionais.

Visto que as UC têm processos de gestão diferentes entre si, é muito difícil classificar os requisitos funcionais pelo benefício genérico para as partes interessadas: o que é crítico numas UC seria apenas útil noutras.

Ao invés, em vez de referirem casos de uso, as diversas partes interessadas inquiridas foram emitindo opinião em relação a grupos de funcionalidades/de casos de uso (cf. secção 3.3).

A partir dessas opiniões e analisando as Tabelas 3 e 4, é prudente e razoável considerar que os grupos de funcionalidades/casos ordenados por ordem de importância são:

1. Gestão de propostas de projetos/estágios;

2. Seleção, Atribuição e Formalização;

3. Gestão de projeto;

4. Marcação de prova.

Por outro lado, a priorização do requisitos não-funcionais é dependente não só das partes interessadas inquiridas, mas também das assunções da equipa de desenvolvimento no início do projeto. O resultado dessa combinação de interesses conduz à seguinte classificação (Tabela 5):

Tabela 5 – Requisitos não funcionais – Benefícios para as partes interessadas

Requisito não funcional	**Classificação**
Funcionalidade	Crítico
Usabilidade	Útil
Confiabilidade	Crítico
Suportabilidade	Crítico
Desempenho	Útil
Requisitos de Design	Importante
Requisitos de Implementação	Importante
Requisitos de interligação	Importante
Requisitos físicos	Importante

3.6 Síntese

Após a descrição da forma como os requisitos foram elicitados e das caraterização das partes interessadas no desenvolvimento do *software*, foi possível sistematizar os requisitos funcionais. Estes requisitos conduzirão a funcionalidades do sistema com interação com os atores do sistema (subconjunto das partes interessadas) e permitirão que o processo de gestão das UC de projeto-estágio do ISEP seja informaticamente suportado e, em parte, automatizado.

Complementarmente, foram caracterizados os requisitos não funcionais, que para além de definirem características que o sistema deverá cumprir, com vista a um melhor desempenho e satisfação dos clientes.

4 Análise de Conceitos de domínio e Processos

Este capítulo tem os seguintes objetivos:

- Descrição e sistematização dos conceitos de gestão das UC de projeto-estágio;
- Análise e sistematização de conceitos de gestão das UC de projeto-estágio.

4.1 Análise de negócio

Esta secção apresenta os modelos de domínio resultantes da sistematização das reuniões descritas na secção 3.1.2 (Figuras Figura 6, Figura 7, Figura 8, Figura 9, Figura 10, Figura 11, Figura 12, Figura 13, Figura 14).

Estes modelos de domínio capturam as descrições de negócio feitas pelo RUC ou diretor de curso, mas não têm a pretensão de serem a representação exata e completa do negócio de cada uma, nem de todas as UC de projeto-estágio dos cursos do ISEP.

Por contraponto, considera-se que são suficientemente representativas da realidade do ISEP, e que permitem perceber a heterogeneidade de abordagens existentes, e dessa forma alertar para as particularidades e dificuldades que daí advêm.

Optou-se por não descrever os diagramas, porquanto se receia que as descrições sejam insuficientes, ambíguas e enganosas, e porque se acredita que a adoção da notação/linguagem UML (OMG, 2019) é semanticamente explicativa e explícita a ponto de não necessitar de qualquer descrição complementar.

4.1.1 **TMDEI – Mestrado em Engenharia Informática**

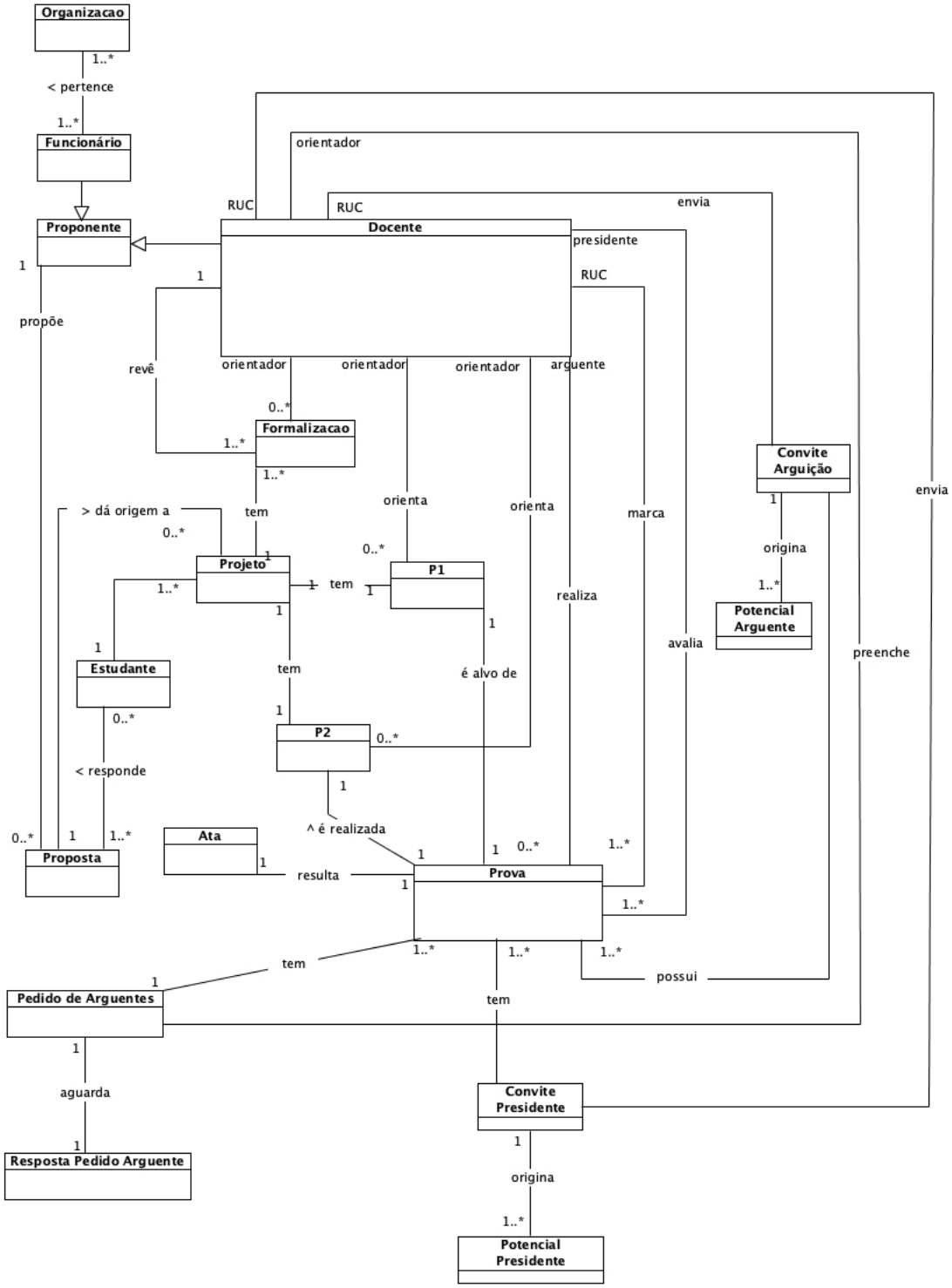

Figura 6 – Modelo de negócio TMDEI

4.1.2 **PESTQ e ESTBI – Licenciatura em Engenharia Química e Licenciatura em Biorrecursos**

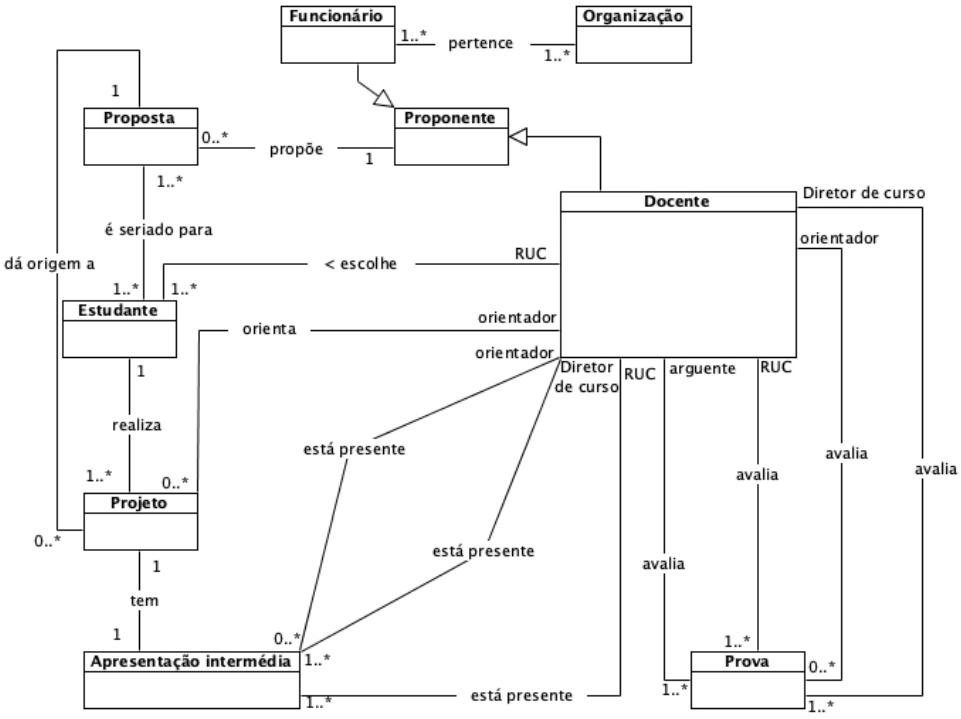

Figura 7 – Modelo de negócio de PESTQ/ESTBI

4.1.3 **DISEST – Mestrado em Engenharia Química**

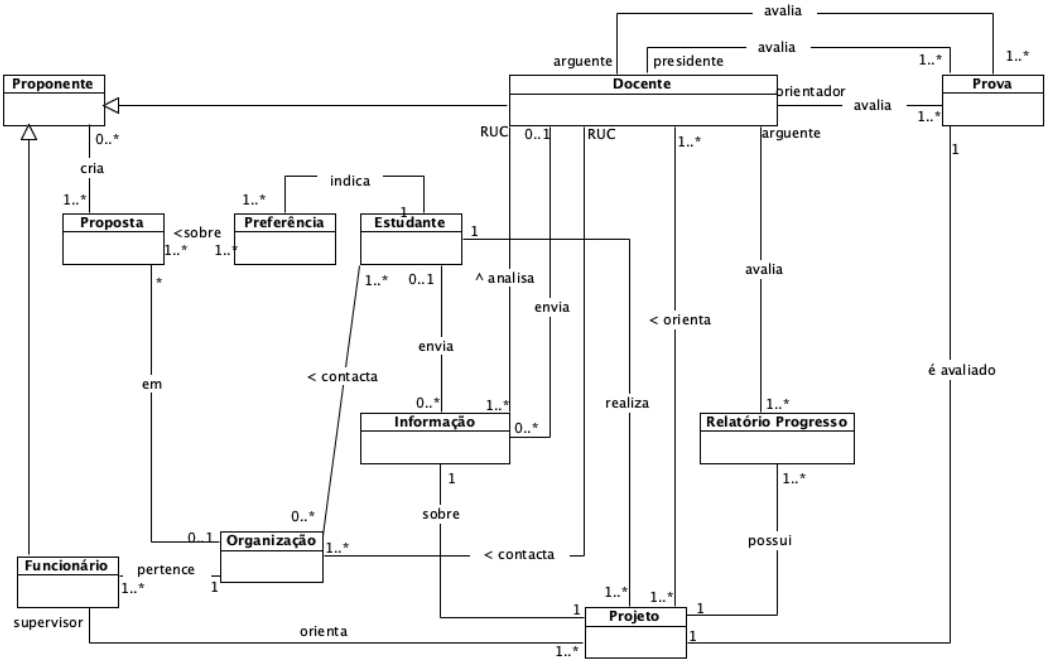

Figura 8 – Modelo de negócio de DISEST

4.1.4 **PESTA – Licenciatura em Engenharia Eletrotécnica e de Computadores**

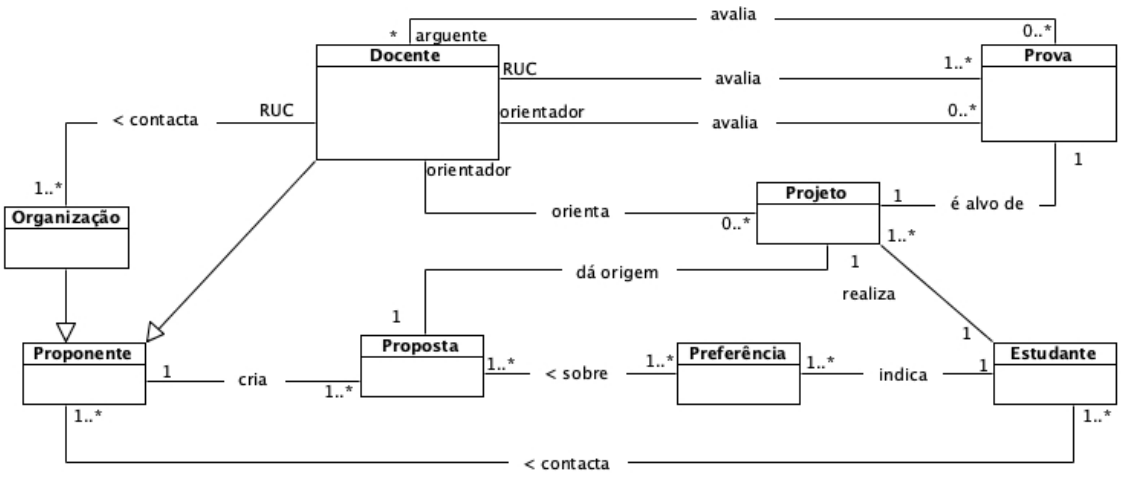

Figura 9 – Modelo de negócio de PESTA

4.1.5 **DIPRE – Mestrado em Engenharia Civil**

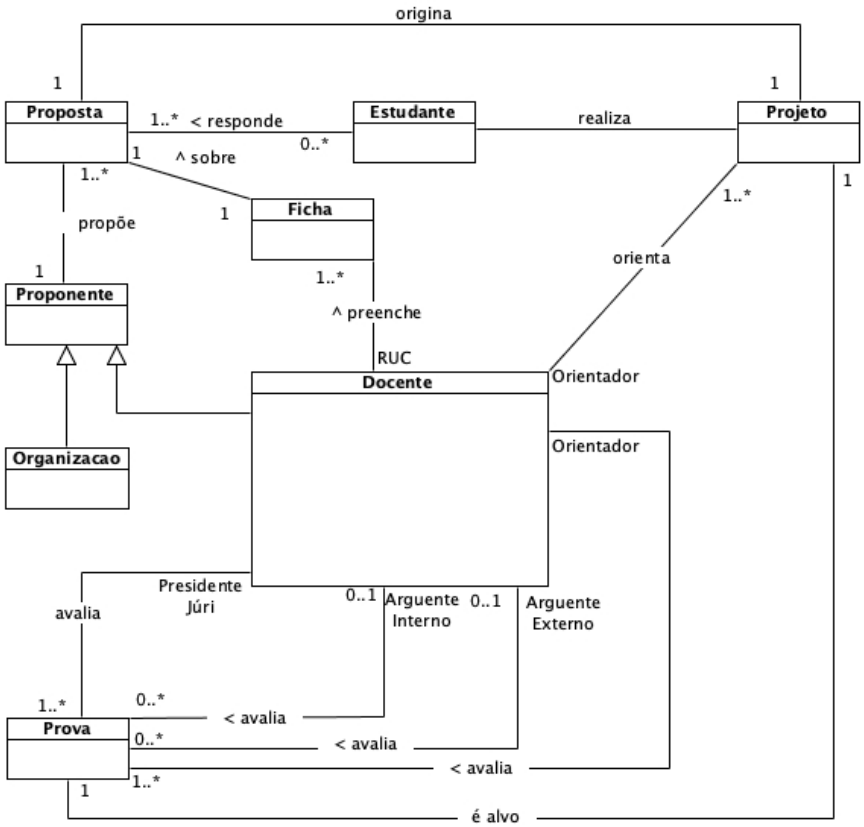

Figura 10 – Modelo de negócio de DIPRE

4.1.6 **PROJI – Licenciatura em Engenharia Civil**

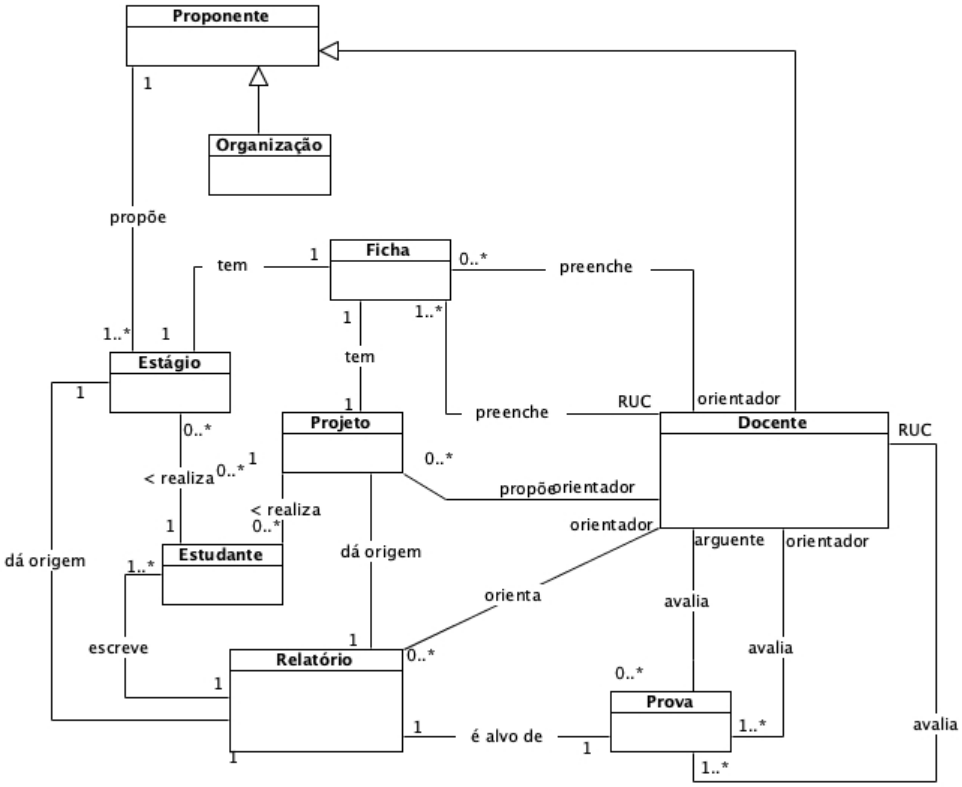

Figura 11 – Modelo de negócio de PROJI

4.1.7 **PESEE – Licenciatura em Engenharia Eletrotécnica – Sistemas Elétricos de Energia**

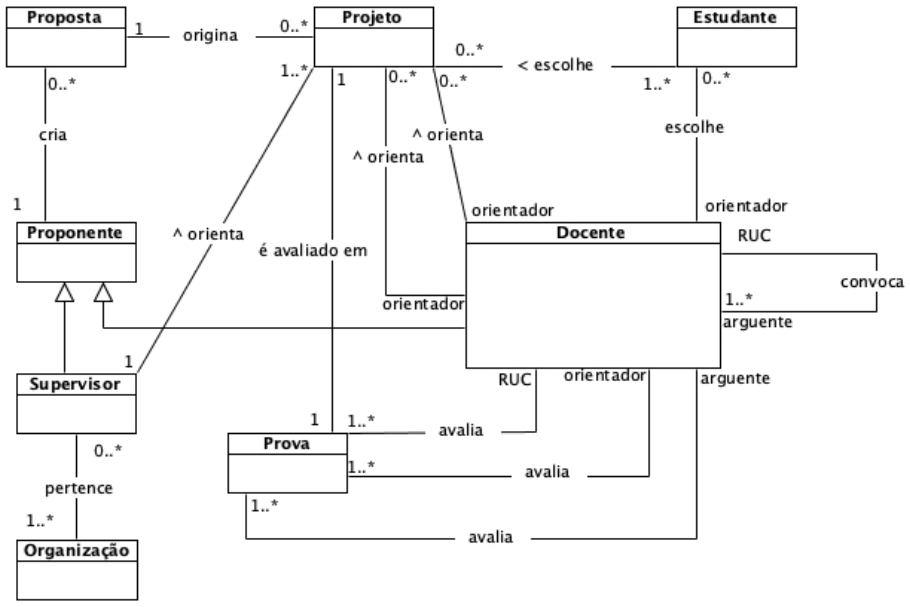

Figura 12 – Modelo de negócio de PESEE

4.1.8 **PESTM – Licenciatura em Engenharia Mecânica**

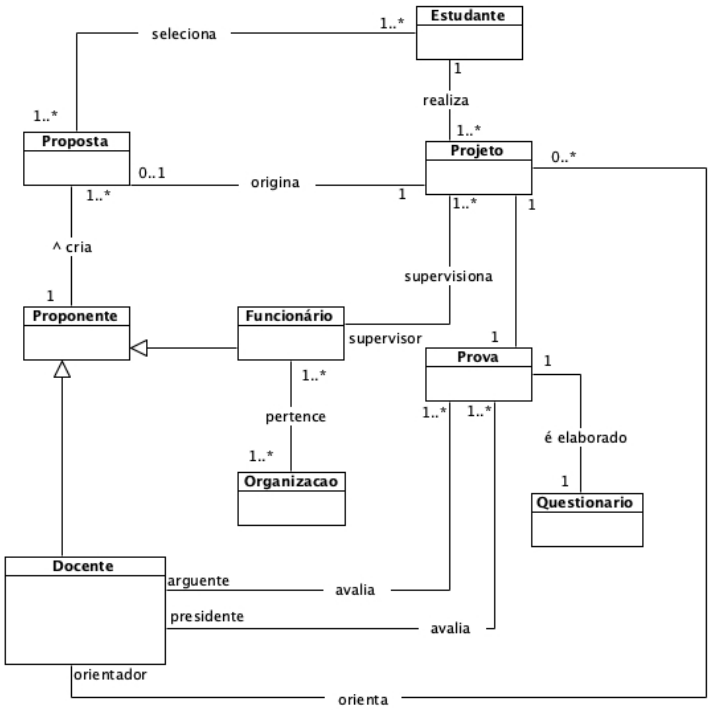

Figura 13 – Modelo de negócio de PESTM

4.1.9 **PROES – Licenciatura em Engenharia de Sistemas**

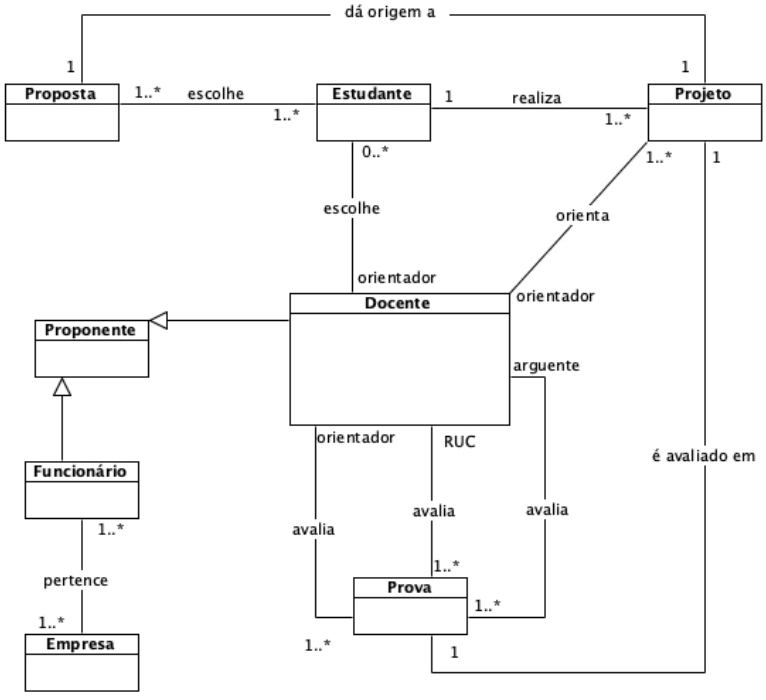

Figura 14 – Modelo de negócio de PROES

4.2 Papéis e Processos

Apesar de terem denominações distintas, os processos de gestão de projetos curriculares das diferentes UC possuem vários conceitos e etapas comuns.

A Figura 15 apresenta os processos/etapas de gestão de todas as UC de forma suficientemente abstrata para que todas as UC estejam representadas.

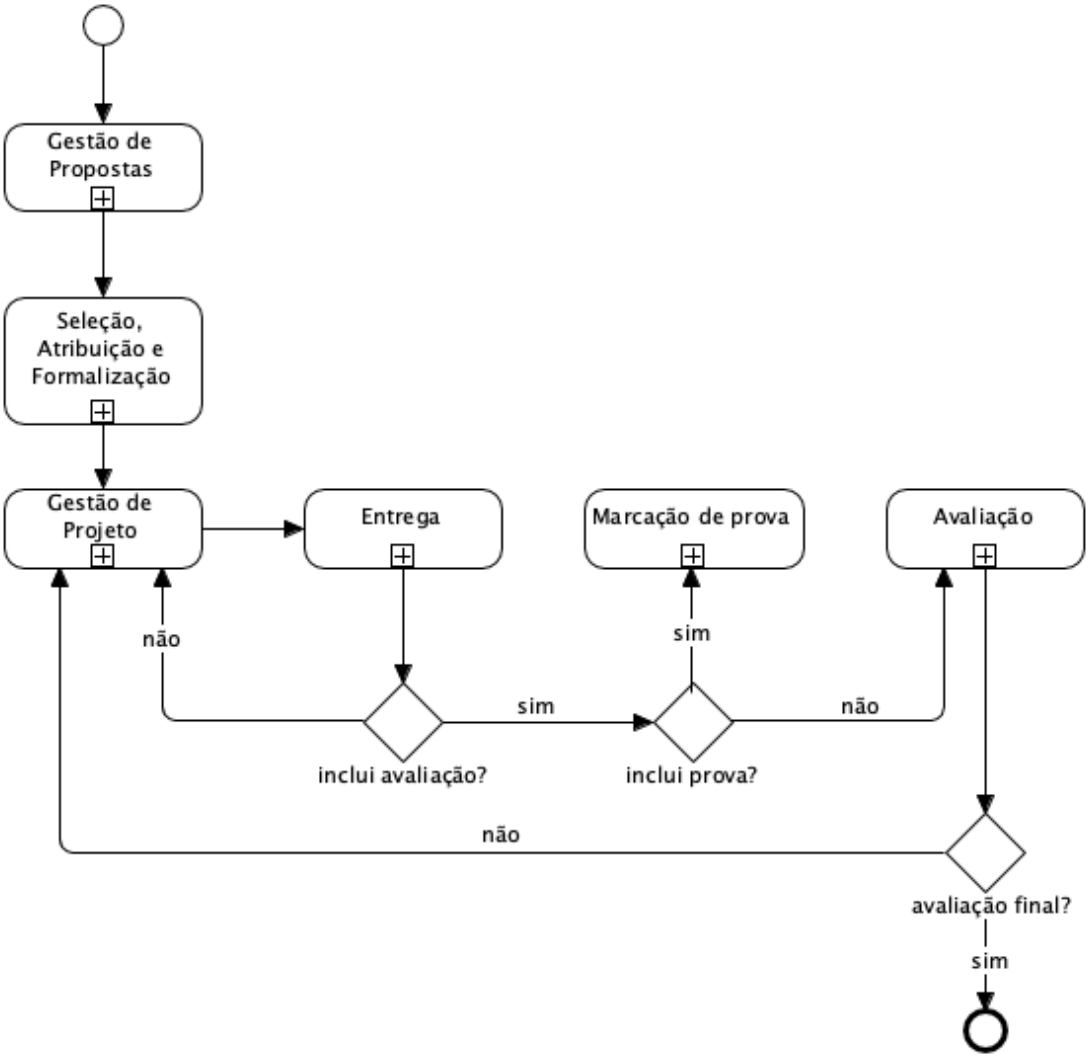

Figura 15 – Organização sequencial das várias etapas

Porque os processo/tarefas variam substancialmente entre UC, não é possível descrever os atores de cada tarefa ou processo, nem tão pouco é adequado representar detalhes sobre cada um dos processos.

4.2.1 **Papéis**

Os utilizadores (cf. Secção 3.2) podem desempenhar duas categorias de papéis no sistema (Figura 17):

- Papéis estáticos são definidos aquando do registo do utilizador no sistema, e são mutuamente exclusivos: administrador, docente, estudante e externo;

- Papéis dinâmicos, são definidos mediante o contexto informativo e funcional. Por exemplo, a função "descarregar tese" estará disponível para docentes e externos (papéis estáticos), que façam parte do júri dessa avaliação, o que tem de ser calculado no momento da invocação da função para cada avaliação.

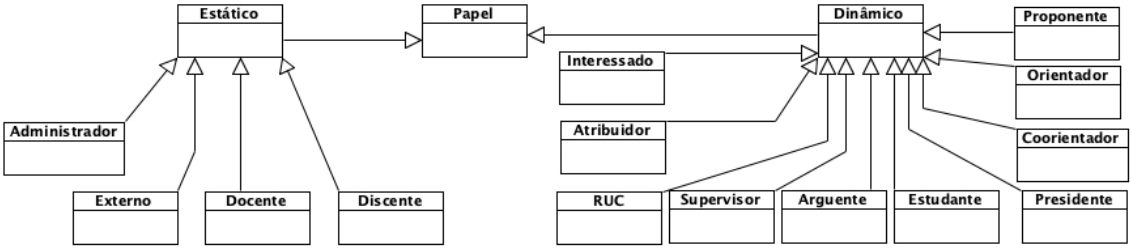

Figura 16 – Diagrama de classes de papéis (notação UML)

Um papel dinâmico é complementar a um papel estático, i.e. um qualquer papel dinâmico é aplicado a um ator com um papel estático. Há papéis dinâmicos que são mutuamente exclusivos entre si em determinado contexto (e.g. presidente e orientador de uma prova não podem ser a mesma pessoa), mas há outros que são complementáveis (e.g. proponente e orientador dum projeto podem ser desempenhados pelo mesmo ator) (

Tabela 6).

Tabela 6 – Exemplo de exclusividade e complementaridade possível entre papéis, no contexto duma mesma proposta-projeto

Papéis		**Estáticos**				**Dinâmicos**								
		Administrador	Docente	Discente	Externo	Proponente	RUC	Interessado	Estudante	Orientador	Coorientador	Supervisor	Arguente	Presidente
Estáticos	Administrador	-	X	X	X	X	X	X	X	X	X	X	X	X
	Docente		-	X	X			X	X			X		
	Discente			-	X	[1]	X			X	X	X	X	X

[1] cf. secção 4.2.2.2.1

	Externo				-	X	X	X	X	X				X
Dinâmicos	Proponente					-	X	[2]	[3]				X	X
	RUC						-		X			X		
	Interessado							-		X	X	X	X	X
	Estudante								-	X	X	X	X	X
	Orientador									-	X	X	X	X
	Coorientador										-	X	X	X
	Supervisor											-	X	X
	Arguente												-	X
	Presidente													-

4.2.2 **Processos**

Nesta secção analisam-se e sistematizam-se processos ou etapas do funcionamento das UC em função dos (grupos de) casos de uso identificados.

4.2.2.1 Gestão de propostas

Esta etapa possui processos distintos nas diferentes UC, mas em todas visa identificar e caraterizar as propostas de projetos.

Enquanto que em algumas UC são as empresas que procuram o contacto com o ISEP com vista à apresentação/divulgação do projeto, noutras UC são os RUC que contactam diretamente as empresas para que estas forneçam estágios aos alunos. Nalguns cursos os estudantes têm a possibilidade de realizar um projeto por eles proposto ou numa empresa que estes contactam diretamente. Na maioria dos casos este contacto existe através de vias não formais (e.g. telefone, email).

A divulgação das propostas criadas pelas empresas também possui muitas diferenças. Alguns RUC mencionam que colocam toda a lista de propostas disponíveis no Moodle ISEP (ISEP, 2019b), outros que enviam as novas propostas aos estudantes via email.

Nalgumas UC existem sites dedicados à publicação das diversas propostas existentes (e.g. GESTProj (DEI, 2019a), GestMEI (DEI, 2019b)). Nestes casos os estudantes podem consultar diretamente as propostas disponíveis.

Muitos dos docentes inquiridos recebem as propostas enviadas pelas empresas externas e pelos outros docentes via email e, consequentemente processam esta informação em Excel, publicitando estas propostas no *Moodle*.

[2] cf. secção 4.2.2.2.1
[3] cf. secção 4.2.2.2.1

4.2.2.2 Seleção, Atribuição e Formalização

Esta etapa visa o processo de escolha e atribuição de um projeto a um estudante, e nalgumas UC, a atribuição de orientador ao projeto/estudante.

4.2.2.2.1 Seleção e Atribuição

Existem algumas diferenças entre os diversos cursos, nomeadamente no que diz respeito à forma como os estudantes selecionam e lhes são atribuídas as propostas:

- Os estudantes selecionam um subconjunto de propostas e é o RUC que atribui a proposta, mediante um conjunto de regras e critérios.
- Os estudantes antes de se candidatem a propostas são seriados pelo RUC. Apenas os estudantes selecionados podem entrar em contacto com o proponente.
- Os estudantes não selecionam propostas e é o RUC que decide qual o estudante mais indicado para uma determinada proposta.
- Os estudantes selecionam as propostas e entram em contacto direto com o proponente. É o proponente que decide qual o estudante que realizará a proposta.

O processo de escolha/atribuição do orientador do projeto também é variável, nomeadamente:

- O RUC atribui um orientador mediante o tema do projeto;
- O estudante tem liberdade para escolher o orientador;
- A proposta já inclui o orientador definido.

4.2.2.2.2 Formalização

Apesar de nomes distintos por UC e de não existir explicitamente nalgumas, o processo de formalização do projeto-estágio consiste em formalizar o acordo entre as partes envolvidas: estudante, proponente e orientador.

No caso de estágio, obriga ainda ao preenchimento e assinatura do protocolo de estágio, e envolve o estudante, organização acolhedora do estágio e a Presidência do ISEP. Este protocolo tem um âmbito de aplicação essencialmente administrativo (e.g. seguro escolar) e quase nenhum âmbito académico-pedagógico.

Nalguns outros casos existe outro processo de formalização em que são descritos detalhes do projeto a realizar como (i) o problema, (ii) os objetivos e (iii) a abordagem a desenvolver no projeto-estágio, ou ainda a forma como os diversos objetivos académicos podem/serão alcançados.

Muitos dos RUC recebem esta formalização via email e processam esta informação em folhas Excel, sendo que alguns destes RUC consideram que é importante o desenvolvimento de um mecanismo que permita fazer a revisão desta formalização, permitindo a inserção de anotações e o posterior envio desta ao estudante.

4.2.2.3 Gestão de projeto

Esta etapa visa a gestão do projeto ao longo do período académico por parte do orientador. No caso de estágio, o supervisor e organização acolhedora são também partes interessadas.

O processo de gestão de cada projeto é muito dependente do orientador e supervisor e organização (caso se aplique), não tendo sido mencionados pelos RUC processos formais de gestão de projeto que necessitem de apoio à gestão centralizada e supervisionada por este software.

4.2.2.4 Entregas

Como resultado dos mecanismos de elicitação de requisitos utilizados, foi possível concluir que existem diversos tipos de entregas nas diferentes UC. Desta forma, as secções seguintes permitem apresentar as diferenças encontradas.

4.2.2.4.1 Entrega final

Sendo que a legislação obriga à submissão de documento de relatório final no Portal do ISEP no que diz respeito aos Mestrados, nas Licenciaturas pode existir alguma flexibilidade neste processo. Apesar de, a maioria das UC funcionar com submissão final no Portal, existem também algumas que permitem esta entrega no *Moodle*, e permitem ou requerem até a entrega dum número fixo de cópias do documento impresso.

Em algumas UC, existem prazos estipulados para esta entrega, que os estudantes têm conhecimento logo no início do ano/semestre, mas noutras UC os alunos apenas conhecem um intervalo em que devem proceder a esta submissão. Estas diferenças foram transmitidas, por alguns docentes, como dificuldades na gestão mais rápida ou mais numerosa de todas estas entregas para que as apresentações sejam realizadas no prazo estipulado pelo ISEP (Anexo B – Atas das reuniões realizadas com os RUC).

4.2.2.4.2 Entregas intermédias

Para além da entrega final, algumas UC definem ainda outras entregas (intermédias ou parciais), que são posteriormente processadas de forma diferente consoante a UC:

- Nalgumas UC não há qualquer entrega periódica ou intermédia;
- Noutras, existem entregas intermédias (e.g. em PESTQ e DISEST), sendo da responsabilidade única do orientador a sua gestão.
- Noutras ainda, há entregas intermédias, e são geridas pelo RUC, caso de TMDEI e PESTI.

No caso das entregas intermédias, estas têm diferentes utilidades e processamentos por UC:

- Não têm qualquer utilidade ou consequência, e tendem a ser apenas indicativas de boas práticas sugeridas a estudantes e orientadores;
- Podem servir para controlo semiformal, podendo ser usadas na avaliação final como clarificação da nota a atribuir.
- Servem para sessão semiformal, não resultando daí qualquer avaliação, mas apenas indicações de melhoria e correção.
- Servem para avaliação formal em prova específica para o efeito.

4.2.2.5 Marcação de prova

A marcação de provas consiste no agrupamento de informação relacionada como o júri e a data da avaliação da prova. Para dar início a este processo é necessário descarregar do Portal ISEP (ISEP, 2019a) ou do *Moodle ISEP* (ISEP, 2019b) as submissões de provas. Seguidamente é necessário proceder à marcação da prova, que varia entre as UC:

- Em algumas existe o convite formal do presidente da prova e do arguente, sendo que o processo só é concluído quando todos aceitaram a proposta;
- Noutras UC, é o RUC que desempenha o papel de presidente de prova, sendo também este a nomear o arguente.

Após o processo de escolha do júri, é necessário determinar o dia, hora e local onde se irá realizar a prova. Este passo pressupõe que seja encontra uma sala livre e que todos os elementos envolvidos na prova tenham disponibilidade. Esta marcação deve ser efetivada no Portal ISEP (ISEP, 2019a).

Contudo, mormente nas licenciaturas, os processos de marcação das provas são bastante expeditos. Em quase todas não há convite ao arguente, e o presidente do júri é o RUC ou alguém duma equipa relativamente pequena (~3 docentes) por ele definida. Na grande maioria dos casos, este processo não carece de grande comunicação com os elementos do júri, sendo antes feita uma comunicação aos mesmos ou ao departamento, que comunica as provas aos docentes. Há ainda casos em que é o orientador que define o arguente e faz a marcação de provas.

4.2.2.6 Avaliação

A elaboração de atas ou a resposta a questionário por parte dos diversos intervenientes são as principais tarefas a realizar na fase de avaliação, sendo que nestes pontos os docentes não apresentaram grandes dificuldades. Apesar de muitos deles necessitarem de processar os resultados manualmente, os RUC contactados não referem isso como um ponto a melhorar/suportar.

4.3 Análise Funcional

A análise funcional tem como objetivo sistematizar e priorizar as funcionalidades do produto a desenvolver (Rich and Holweg, 2000) (Anexo D e Anexo E) . Para tal, será usado o método QFD (cf. Secção 4.3.1).

O *Quality Function* Deployment (QFD) é um método que permite analisar os requisitos de qualidade e identificar quais os que possuem maior relevância para o cliente (Roberts, 2007).

Como forma de sistematizar esta análise, será utilizada a casa da qualidade. Esta abordagem tem como objetivo representar a hierarquia das prioridades das partes interessadas e nomeadamente dos RUC (cf. secção 3.5) e as características de qualidade que o software deve suportar ou melhorar.

Para tal, a casa da qualidade (Hauser and Clausing, 1988) permite atingir esse objetivo correlacionando:

- Requisitos/necessidades/desejos dos clientes (*Whats*), priorizados na secção 3.5 como:
    - Gerir proposta;
    - Seleção, atribuição e formalização;
    - Gestão de projeto;
    - Marcação de prova.
- Requisitos do produto (*Hows*), classificados na secção 3.5 como:
    - Funcionalidade;
    - Confiabilidade;
    - Suportabilidade.

A Figura 17 representa a casa da qualidade construída no âmbito do projeto referido.

Legend		
Θ	Strong Relationship	9
O	Moderate Relationship	3
▲	Weak Relationship	1
++	Strong Positive Correlation	
+	Positive Correlation	
−	Negative Correlation	
▼	Strong Negative Correlation	
▼	Objective Is To Minimize	
▲	Objective Is To Maximize	
x	Objective Is To Hit Target	

Row #	Max Relationship Value in Row	Relative Weight	Weight / Importance	Demanded Quality (a.k.a. "Customer Requirements" or "Whats") / Quality Characteristics (a.k.a. "Functional Requirements" or "Hows")	1 Funcionalidade	2 Confiabilidade	3 Suportabilidade
				Column #	1	2	3
				Direction of Improvement: Minimize (▼), Maximize (▲), or Target (x)	x	x	x
1	9	25,0	25,0	Gerir proposta	Θ	▲	Θ
2	9	25,0	25,0	Selecção, Atribuição e Formalização	Θ	▲	Θ
3	9	25,0	25,0	Gestão de projeto	Θ	▲	Θ
4	9	25,0	25,0	Marcação de prova	Θ	O	Θ
				Target or Limit Value	Supported	Supported	Supported
				Difficulty (0=Easy to Accomplish, 10=Extremely Difficult)	7	7	7
				Max Relationship Value in Column	9	3	9
				Weight / Importance	900,0	150,0	900,0
				Relative Weight	46,2	7,7	46,2

Figura 17 – Casa da qualidade do projeto

Observando os pesos relativos calculados para cada requisito (última linha da casa da qualidade da Figura 17) é possível concluir que os requisitos mais importantes são: a Funcionalidade (46,2%) e a Suportabilidade (46,2%), pelo que devem ser tidos em especial consideração aquando do desenho do sistema a desenvolver.

4.4 Síntese

Baseado nos resultados da elicitação descritos no capítulo anterior, analisou-se e sistematizou-se o negócio como um conjunto ordenado de etapas concetuais comuns entre UC, mas que no detalhe, são diferentes e evolutivas.

Apresentou-se ainda uma caraterização dos papéis e dos processos/etapas de negócio e priorizaram-se analiticamente os requisitos.

5 Design

Neste capítulo descreve-se o design do sistema considerando os requisitos e contexto de negócio apresentados nos capítulos anteriores, detalhando alternativas e decisões.

Começa-se por analisar e decidir a abordagem concetual a adotar, o que vai condicionar a arquitetura do sistema.

Passa-se depois, então, à análise e design arquitetural. Para a sua descrição adota-se uma abordagem baseada na combinação de dois modelos:

- Modelo C4 (Brown, 2011), do qual será adotado a proposta de 4 níveis de abstração/foco/zoom diferentes:
    - Nível 1, descreve o contexto (abstração/granularidade) de sistema;
    - Nível 2, descreve o contexto de *container*[4] (contentor);
    - Nível 3, descreve o contexto de componente;
    - Nível 4, descreve o contexto de código.
- Modelo *View and Beyond* (V&B) (Bachmann et al., 2010) que propõe três tipos de vistas:
    - Vista de módulos, descreve os componentes de implementação de um sistema
    - Vista de componentes e conetores (C&C), onde são representadas as unidades de execução do sistema
    - Vista de alocação, relaciona o sistema com os ambientes de desenvolvimento e execução

[4] Algo que contém código ou dados (e.g. base de dados, aplicação móvel, aplicação web cliente).

A descrição começa pelo nível de maior abstração (nível 1) para o de menor abstração (nível 4), e em cada nível apresentam-se os vários tipos de vistas V&B adotadas de acordo com as necessidades de comunicação e debate das alternativas em causa.

O capítulo termina com a apresentação da priorização dos requisitos a implementar, considerando agora, também o impacto arquitetural.

5.1 Alternativas concetuais

Adotando o modelo NCD (Koen et al., 2004), a terceira etapa da análise de valor pressupõe a definição de diversas alternativas (concetuais), que permitirão dar resposta ao problema e objetivos enunciados (Rich and Holweg, 2000).

5.1.1 Identificação de alternativas

Desta forma, apesar das ideias terem sido discutidas com RUC de diversas UC, estas têm formas de funcionamento diferentes, pelo que será necessário que o *software* produzido suporte estes diferentes funcionamentos.

Assim, foram enunciadas as seguintes alternativas:

- Alternativa A – Desenvolver um software para cada uma das UC;
- Alternativa B – Desenvolver um software que seja configurável e, assim, ser possível utilizá-lo nos diferentes processos de gestão dos projetos curriculares;
- Alternativa C – Desenvolver um software único que unifique os diferentes processos, para que utilizem uma abordagem única de negócio.

Realizar-se-á um estudo mais detalhado destas alternativas, de forma a validar estas afirmações e assim escolher a melhor alternativa a adotar. Para tal, será aplicado o método de decisão multicritério AHP.

5.1.2 Definição de critérios

Tendo como base a análise da priorização dos requisitos (cf. secção 3.5) e da análise de qualidade do sistema a desenvolver (cf. secção 4.3), os critérios de decisão das alternativas são:

- Realização dos requisitos funcionais mais importantes para o cliente (R) (Gestão de propostas, Seleção, Atribuição e Formalização, Gestão de projeto, Marcação de prova) (cf. secção 3.5);

- Funcionalidade (F), relacionada sobretudo com Autenticação, Autorização e Auditoria (cf. secção 3.4.1);

- Suportabilidade (S), relacionada com a qualidade do software (cf. secção 3.4.4).

5.1.3 **Decisão multicritério**

Um método de decisão multicritério assenta em técnicas numéricas, que ajudam quem os aplica a escolher a melhor opção de entre as alternativas discretas enunciadas. Estes métodos são aplicados a partir do cruzamento das alternativas com diversos critérios.

O método de decisão multicritério utilizado é o *Analytic Hierarchy Process* (AHP) (Saaty, 2008). Este método permite a utilização de critérios qualitativos e quantitativos no processo de avaliação, e desta forma, separar o problema da decisão em níveis hierárquicos, melhorando a compreensão e avaliação do processo.

De acordo com os cálculos efetuados (cf. Anexo F – Método de Análise Hierárquica – AHP), é possível concluir que em termos de análise da funcionalidade, todas as alternativas são possíveis, pelo que terá de ser analisada a suportabilidade de cada uma das alternativas.

A alternativa A pressupõe o desenvolvimento de várias aplicações, já é colocada de parte a utilização da configurabilidade, incluída na suportabilidade. A alternativa C, não permite que as UC tenham funcionamentos diferentes, logo não existe a necessidade de utilizar a configurabilidade, incluída na suportabilidade. O facto de a alternativa C não permitir funcionamentos distintos, também não permite dar resposta aos requisitos funcionais desejados pelo cliente. Dessa forma, a alternativa B é a única que permite suportar os três critérios avaliados.

5.2 Design arquitetural de sistema (Nível 1)

Inicia-se a descrição do sistema por uma vista de abstração completa dos detalhes do sistema a desenvolver, concentrando-se no contexto tecnológico envolvente (Figura 18).

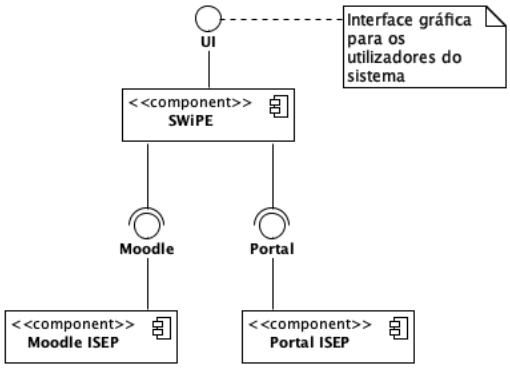

Figura 18 – Vista arquitetural de sistema contextualizado

Depois de tudo o que já foi descrito neste capítulo, este nível de abstração serve apenas como representação base a partir da qual o sistema SWiPE é progressivamente descrito em níveis progressivamente menos abstratos (Nível 2 e Nível 3).

5.3 Design arquitetural de contentores (Nível 2)

No design arquitetural a este nível de abstração, para suportar os requisitos funcionais identificados (cf. secção 3.3), é fundamental ter em conta algumas dimensões ortogonais enunciadas como requisitos não funcionais (cf. secção 3.4), nomeadamente:

- Aplicação multiutilizador;
- Autenticação;
- Autorização;
- Rastreabilidade de informação e ações.

5.3.1 **Vista Components & Connectors**

A vista Components & Connectors (C&C) descreve a forma como o sistema se encontra estruturado, do ponto de vista de partes/componentes e as suas interações.

5.3.1.1 Arquitetura base

Para dar início ao design arquitetural, e partindo da assunção de que se pretende uma aplicação multiutilizador, começou por se separar a responsabilidade da gestão de conceitos de negócio da interação com o utilizador.

Com o objetivo de criar um sistema centrado nos conceitos e ações de negócio e suportar múltiplos utilizadores simultâneos, decidiu-se adotar uma arquitetura que combina dois estilos arquiteturais complementares:

- Cliente-Servidor, na qual existe uma separação da responsabilidade de acesso aos conceitos de negócio, da interação com o utilizador (Saternos et al., 2014);

- Arquitetura Orientada a Serviços (*Service-Oriented Architecture* (SOA)), que permite desenvolver sistemas independentes e modulares, nos quais os serviços disponibilizados são especificados por interfaces que são implementadas e executadas em ambientes mais ou menos indiscriminados (IBM, 2014c). Neste projeto, porque se assume o requisito de desenvolver uma aplicação web, pressupõe a adoção de recomendações e padrões arquiteturais potenciadores da solução, como seja REST ou SOAP.

Combinando estes dois estilos, define-se uma arquitetura para o sistema SWiPE com dois contentores[5] (Figura 19):

- Servidor (SWiPE Backend[6]), responsável pelo cumprimento da lógica de negócio e por providenciar acessos aos dados através de uma interface SOA; no caso em particular, adotar-se-á o padrão *Representational State Transfer*[7] (Fielding, 2000) que potencia um ainda maior desacoplamento entre cliente e servidor que o SOAP, nomeadamente pela adoção de HATEOAS[8] (Fowler, 2010);

- Cliente (SWiPE Frontend[9]), com a responsabilidade de permitir a interação com o utilizador e em função destas, solicitar as informações e comandos ao servidor através da interface REST. Este componente segue uma abordagem arquitetural *Rich Internet Application* (RIA) (Linnenfelser et al., 2010). Uma RIA é executada num navegador web, sendo que não é necessária a instalação por parte do cliente. Para além disso, este tipo de abordagem pressupõe o cumprimento de standard web (Berners-Lee, 1994) e a utilização do protocolo HTTP. O SWiPE FE também é uma *Single Page Application* (SPA),

[5] Uma unidade de instalação e execução separada que executa código ou armazena dados (Brown, 2011).
[6] SWiPE BE ou apenas BE por simplificação.
[7] Representational State Transfer (REST).
[8] Hypermedia As The Engine Of Application State.
[9] SWiPE FE ou apenas FE por simplificação.

uma vez que se deseja que seja carregada apenas uma única página HTML e que esta se altere mediante a interação com o utilizador.

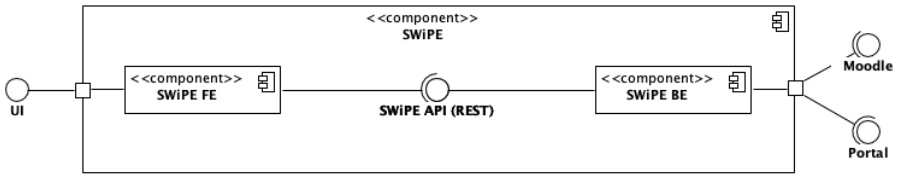

Figura 19 – Design arquitetural de sistema – Inicial (notação UML)

Porque o SWiPE FE e SWiPE BE são sistemas independentes e modulares e cada um com o seu ciclo de vida, constituindo o SWiPE pela interação em tempo de execução, doravante o sistema/componente SWiPE será representado apenas em circunstâncias em que seja vantajoso para a transmissão de ideias.

5.3.1.2 Suporte a autenticação

Após a definição da arquitetura inicial do sistema, foi analisada a dimensão de autenticação (cf. secção 3.4.1.1). Nesta dimensão surgiram diversas alternativas arquiteturais que tiveram de ser analisadas com vista à decisão.:

- Alternativa 1: Desenvolvimento ou adoção de mecanismo de autenticação privado;
- Alternativa 2: Adoção de um mecanismo de autenticação já existente e já adotado no ISEP e pelos atores.

Porque é requisito garantir que os utilizadores confiam no sistema quando inserem as suas credenciais pessoais, optou-se por não adotar um mecanismo de autenticação privado (alternativa 1), pois isso implicaria que os utilizadores tivessem novas credenciais ou que partilhassem as credenciais de acesso com o sistema, e no qual confiassem. Como consequência, a decisão recaiu na adoção da alternativa 2, em que o sistema a desenvolver recorre a um sistema externo de autenticação, que também é usado pelo ISEP e em particular pelo Portal e Moodle: Azure Active Directory (Microsoft, 2019a). Dessa forma, a grande maioria das credenciais dos utilizadores do SWiPE já existem num sistema com o qual os utilizadores já estão acostumados e no qual confiam (Figura 20).

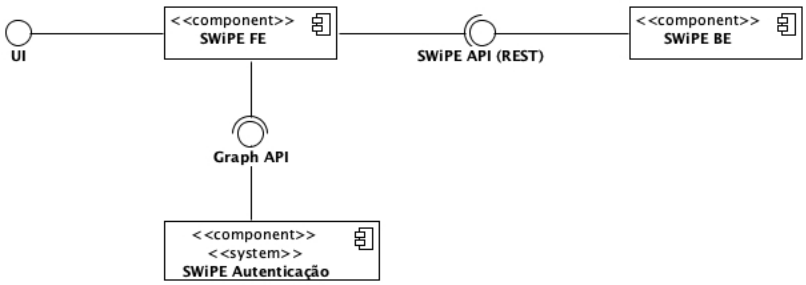

Figura 20 – Design arquitetural do sistema – Autenticação (UML)

Note-se que o Portal e Moodle são apresentados no diagrama anterior com objetivo de facilitar a contextualização, mas não serão futuramente representados.

O processo de autenticação baseado nesta decisão será descrito com mais detalhe nas secções 5.4.3.3 e 5.4.4.

5.3.1.3 Suporte à autorização

O mecanismo de autorização deve ter capacidade de lidar com papéis dinâmicos e estáticos (cf. secção 4.2.1).

O sistema de autenticação proposto anteriormente (bem como a grande maioria dos sistemas de autenticação atuais) são também sistemas de controlo de acesso/autorização. A grande maioria adota uma abordagem de Controlo de Acessos Baseados em Papéis[10] (Microsoft, 2019b), i.e. de papéis estáticos, e algum suporte a papéis dinâmicos (Controlo de Acesso Baseado em Atributos[11] (Hakansson, 2019).

Assim, surgiram duas alternativas de autorização:

- Alternativa 1: Utilizar um sistema de autorização já existente, nomeadamente integrado com o sistema de autenticação proposto anteriormente, que já tem suporte para papéis estáticos, mas com insuficiente capacidade para papéis dinâmicos (Microsoft, 2017). A Figura 21 representa a inclusão desta responsabilidade no contexto da arquitetura anterior;

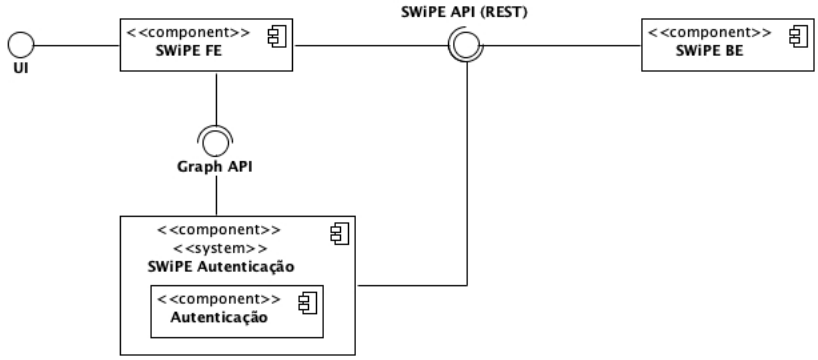

Figura 21 – Design arquitetural do sistema com Autorização – Alternativa 1 (UML)

- Alternativa 2: Desenvolver um sistema de autorização privado em complemento com o sistema de autenticação e autorização baseado em papéis estáticos proposto anteriormente. Este sistema de autorização complementa o sistema de autenticação anterior com a autorização baseada em contexto (i.e. em atributos), acedendo ao

[10] Role-Based Access Control (RBAC).
[11] Attribute-Based Access Control (ABAC).

contexto necessário, no âmbito do SWiPE Backend. A Figura 22 representa a inclusão desta responsabilidade no contexto da arquitetura anterior.

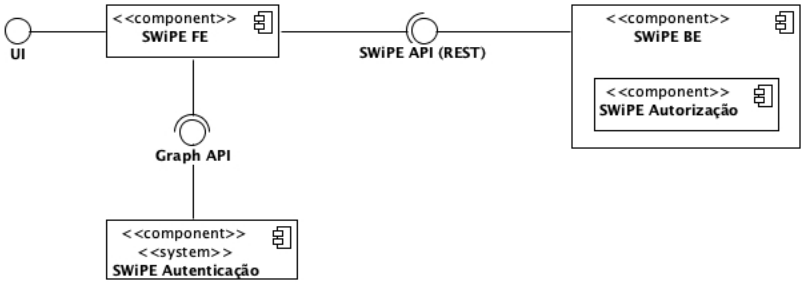

Figura 22 – Design arquitetural de sistema com Autorização – Alternativa 2 (UML)

Considerando as limitações dos sistemas de autorização existentes, nomeadamente do *Azure Active Directory*, em suportar extensivamente ABAC, decidiu-se desenvolver um mecanismo de autorização privado correspondente à alternativa 2.

5.3.1.4 Rastreabilidade de informação e ações

O objetivo é que a qualquer momento seja possível consultar o histórico de uma determinada entidade e saber que utilizadores realizaram ações na mesma. Para isso, é necessário prover as responsabilidades de criação, armazenamento e consulta das ações nalguma parte do BE.

Este é um cenário típico de adoção complementar dos seguintes padrões:

- Domain event, para capturar as alterações no estado da aplicação (Fowler, 2005a). Estes eventos transportam alguma informação relevante sobre a ação ocorrida, nomeadamente (i) ação realizada, (ii) informação resultante, (iii) utilizador que realizou a ação e (iv) informação adicional (e.g. erros ocorridos);

- *Audit Log*, para que todas as alterações de estado nas entidades sejam registadas (Fowler, 2004);

- *Event Sourcing*, permite o uso dos eventos armazenados para a consulta de informação de negócio ao longo do tempo, e não apenas a informação descritiva do estado atual[12] (Fowler, 2005b).

Assim sendo, as responsabilidades de criação e armazenamento dos eventos podem ser atribuídas de duas formas:

- Alternativa 1: criação e armazenamento e consulta no SWiPE Backend

[12] *Event Sourcing* é frequentemente adotado em contextos e com objetivos diferentes, nomeadamente na adoção de *Domain-Driven Design* e na integração de sistemas orientados por coreografias.

- Alternativa 2: criação no SWiPE Backend e armazenado numa *message store* (Hohpe and Woolf, 2003). A consulta dos eventos pode ser realizada sobre portais disponibilizados tipicamente pelo software de gestão do *message store*, sem qualquer requisito ou funcionalidade excecional.

Aplicando o princípio da *Single Responsability Principle*[13] (Martin, 2000) e *Pure Fabrication* (Larman, 2005), a responsabilidade pela criação deve ser separada do armazenamento, pelo que se opta pela alternativa 2.

Decorrente das várias decisões, o design arquitetural resultante é descrito na Figura 23:

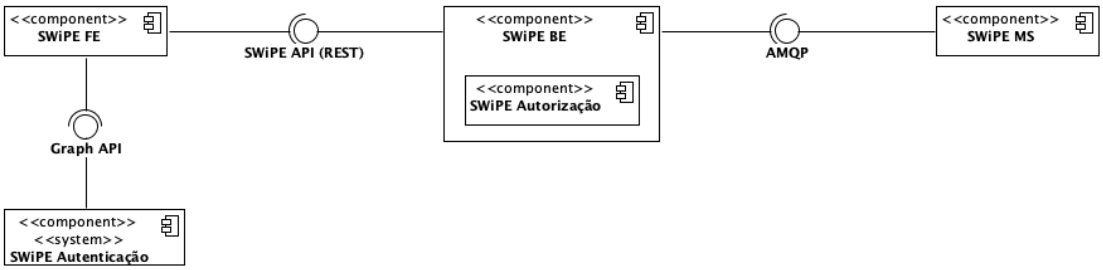

Figura 23 – Design arquitetural de sistema final (UML)

5.3.1.5 Síntese

A Tabela 7 representa as responsabilidades de cada um dos componentes que constituem a arquitetura do sistema.

Tabela 7 – Responsabilidades dos componentes

Componente	**Responsabilidades**
SWiPE FE	- Apresentação da UI ao utilizador - Recorrer às funcionalidades/recursos do SWiPE BE, via API REST - Permitir a autenticação do ator junto do Azure AD, via Graph API
SWiPE BE	- Gerir a informação a gravar na base de dados - Fornecer serviços/recursos através uma API REST - Publicar eventos na *message/event store* via interface AMQP
SWiPE MS	- Armazenar eventos sob pedido, via interface AMQP
Azure Active Directory	- Permitir a autenticação dos atores via Graph API

5.3.2 **Vista Comportamental**

Os diagramas de sequência seguintes, em notação UML, descrevem as funções destes componentes na execução de várias funcionalidades:

[13] Princípio da Responsabilidade Única.

- Autenticação: o utilizador não autenticado pretende aceder à página inicial (Figura 24):

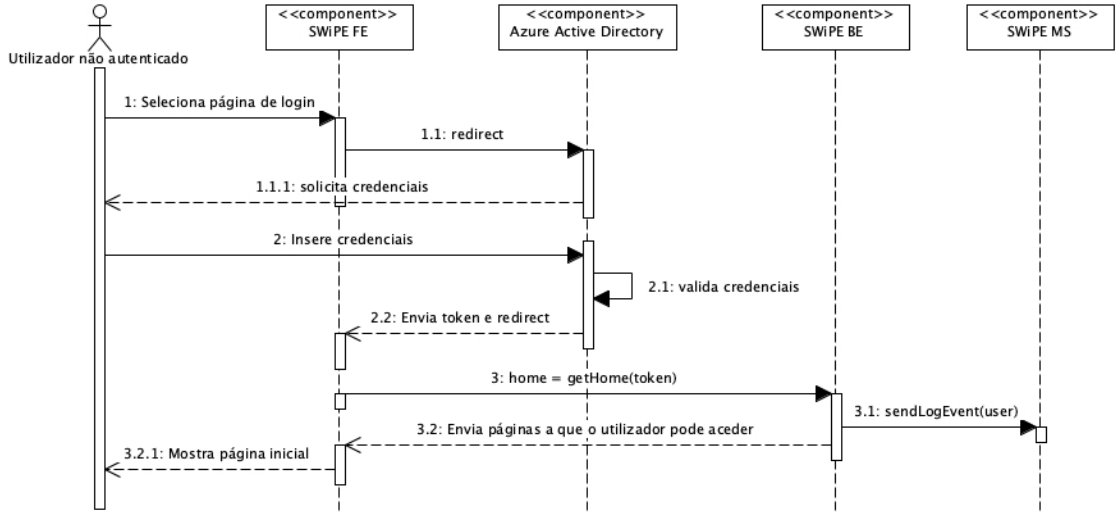

Figura 24 – Autenticação e Apresentação da página inicial (diagrama de sequência UML)

- Criar proposta de projeto e adicioná-la à lista existente (Figura 25):

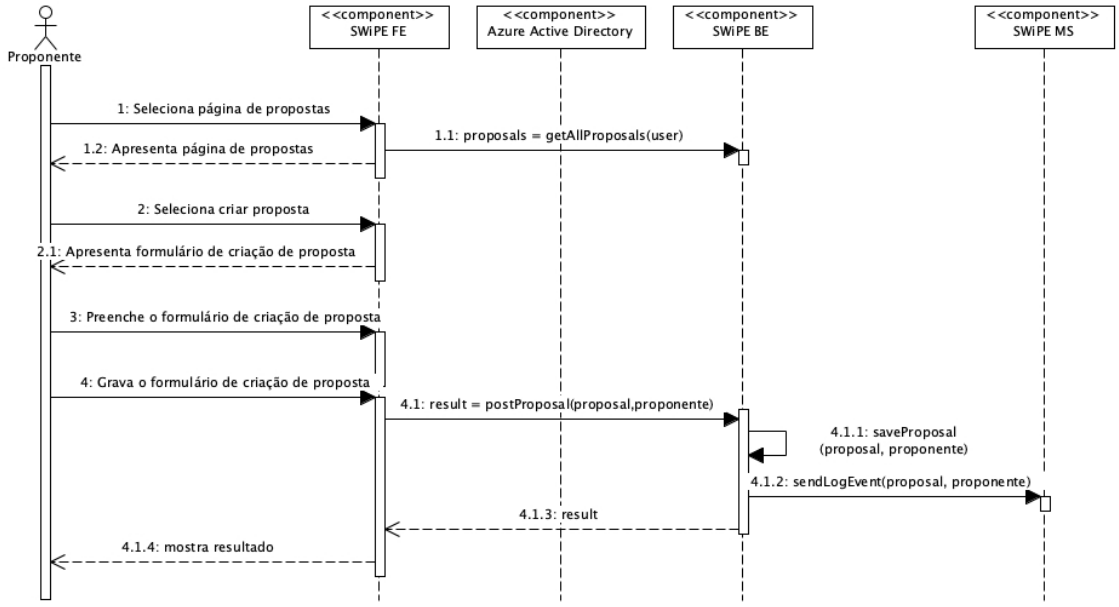

Figura 25 – Diagrama de sequência – Criar proposta

5.3.3 **Vista de Módulos (Uso)**

O sistema é composto por quatro módulos (contentores, na terminologia C4) que se interligam entre si por relações de "uso", e.g. SWiPE Frontend usa SWiPE Backend:

- SWiPE Backend, correspondendo direta e completamente ao componente SWiPE Backend descrito na vista C&C, que faz uso de SWiPE Message Store;

- SWiPE Frontend, correspondendo direta e completamente ao componente SWiPE Frontend descrito na vista C&C, que faz uso do SWiPE Backend e do SWiPE Autenticação;

- SWiPE Autenticação, correspondendo direta e completamente ao componente SWiPE Autenticação descrito na vista C&C;

- SWiPE Message Store, correspondendo direta e completamente ao componente SWiPE Message Store descrito na vista C&C.

A Figura 26 representa em diagrama de pacotes em notação UML da descrição anterior.

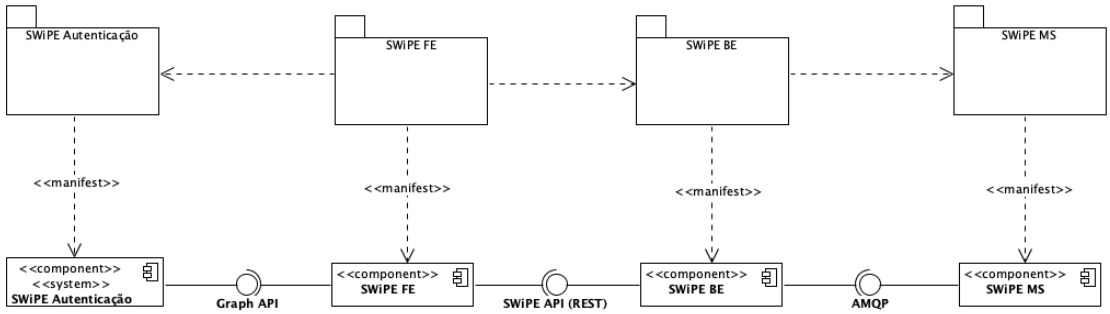

Figura 26 – Vista de módulos do sistema

Cada um dos módulos anteriores é um projeto de software distinto, sendo responsabilidade deste projeto:

- construção do módulo SWiPE BE;

- construção do módulo SWiPE FE;

- configuração do módulo SWiPE MS;

- configuração do módulo SWiPE Autenticação.

5.3.4 **Vista de Alocação**

A vista de alocação foca-se na forma como os componentes do sistema são instalados (Kruchten, 1995, p. 1). Desta forma, a Figura 27 complementar as vistas de módulos e C&C anteriores.

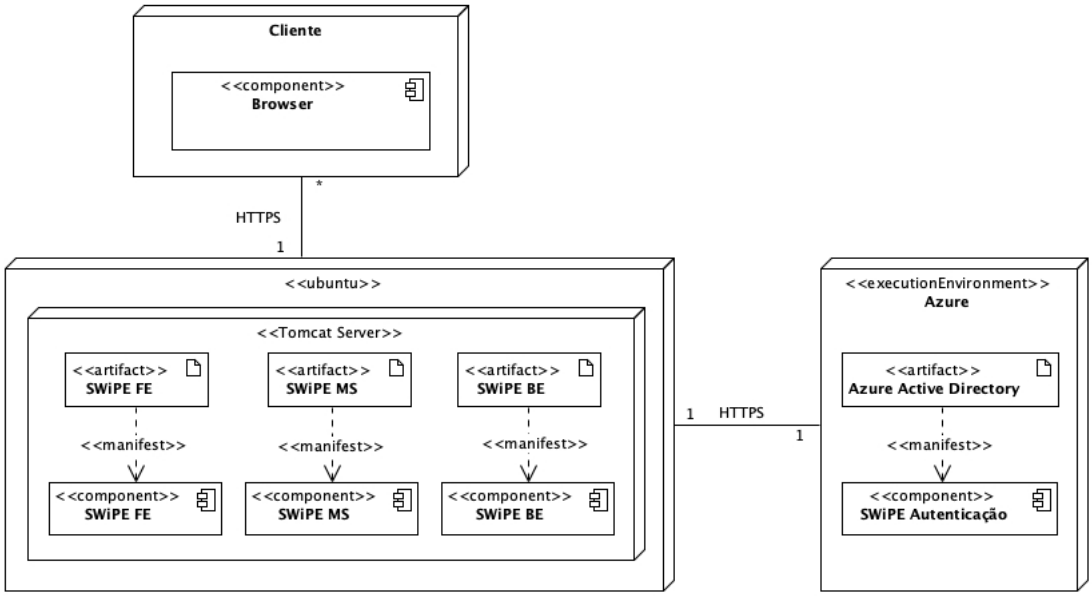

Figura 27 – Vista de alocação

Note-se que, embora o artefacto SWiPE FE esteja alocado no nó servidor, sendo uma Single Page Application web, é executado no navegador web (browser) do nó cliente.

5.4 Design arquitetural de SWiPE Backend (Nível 3)

Depois de tomadas as decisões no que diz respeito ao design arquitetural do sistema, foi necessário definir a arquitetura de cada uma das aplicações. Nas secções seguintes, será apresentado o design arquitetural do SWiPE Backend.

5.4.1 **Vista C&C**

No design arquitetural do SWiPE BE, e considerando a grande quantidade de responsabilidade que lhe estão atribuídas, optou-se por adotar um estilo arquitetural por camadas (Fowler, 2002). Contudo, o estilo é suficientemente abstrato para contemplar diversas e pouco adequadas arquiteturas.

Optou-se então por adotar o padrão *Onion Architecture* (Palermo, 2008) que tendo um estilo em camadas, determina as responsabilidades das camadas e aplica-se adequadamente a

aplicações de gestão de informação de negócio, pois é centrado no modelo/domínio e no estabelecimento de interfaces entre camadas (Figura 28).

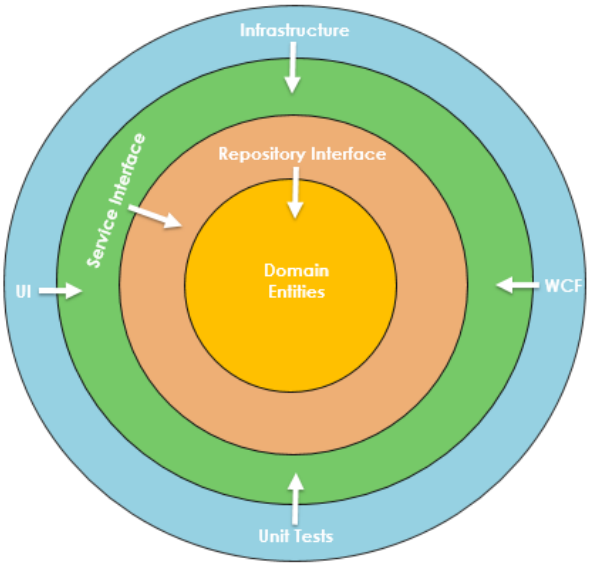

Figura 28 – Representação visual da *Onion Architecture*[14]

A *Onion Architecture* permite, por isso, desenvolver aplicações modulares, pois o contrato de interface permite mudar a implementação dos módulos sem interferências no acoplamento. Pelo mesmo raciocínio, também potencia a realização de testes, pois é possível o teste a cada camada isoladamente, pela substituição das camadas de suporte ao teste por *mocks*, que implementam a interface mas não a lógica subjacente a cada camada, não correndo assim o risco de ver os testes a uma camada infetados pela lógica de outra (Palermo, 2008).

[14] Fonte: https://www.codeguru.com/imagesvr_ce/2236/Onion1.png

A Figura 29 representa o design arquitetural do SWiPE BE pela adoção dos princípios da *Onion Architecture*.

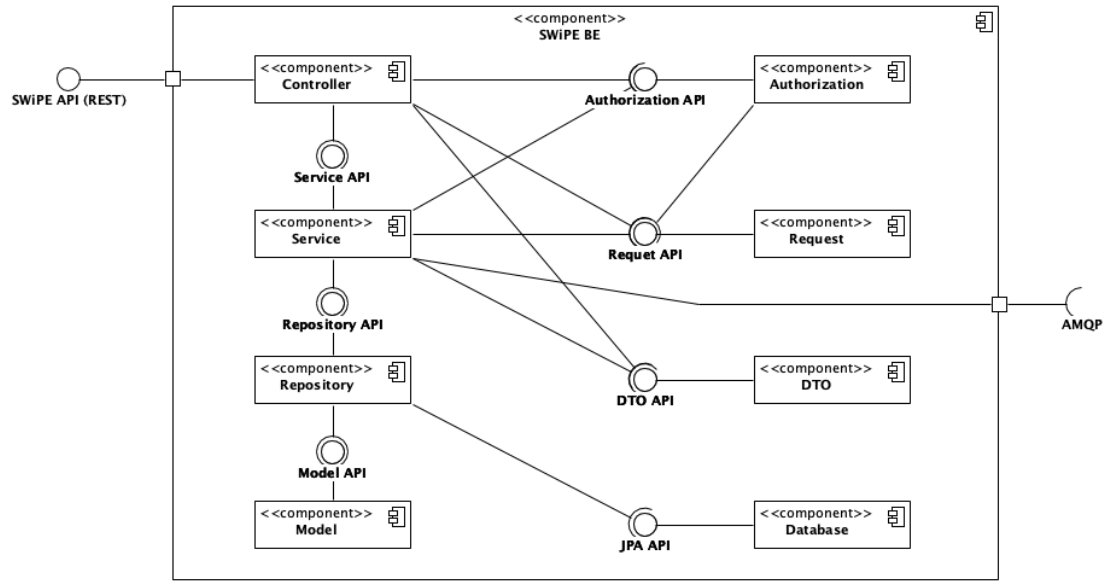

Figura 29 – Design Arquitetural de Aplicação – BE

Os componentes do lado esquerdo do componente adotam as responsabilidades concetualmente definidas pelo padrão *Onion Architecture* mas é importante descrever a sua aplicação ao projeto:

- Controller, desempenha a função de interface de entrada de pedidos à aplicação, delegando as funcionalidades noutros componentes, nomeadamente Service e Authorization;

- Service, desempenha a responsabilidade da camada de aplicação (*service interface*), disponibilizando funcionalidades de negócio, recebendo e devolvendo a informação de domínio/modelo na forma de DTO. Para além disso, é responsável por:

    - garantir o cumprimento de autorizações de acesso e manipulação de dados, requerendo essa responsabilidade ao componente Authorization;

    - publicar os eventos de domínio para o SWiPE MS via a interface AMQP, respondendo ao requisito de rastreabilidade;

- Repository é um conjunto de classes responsáveis por (i) gerir a persistência dos objetos (OO) na base de dados (e.g. relacional), e pela (ii) disponibilização de funções de query e manipulação de listas de objetos das classes de domínio/modelo (OO);

- Model (ou Domain), é o conjunto de classes necessárias para a gestão da informação de negócio. São classes de software que representam conceitos de negócio (cf. secção

4.2), bem como outras classes necessárias à gestão de informação relevante numa perspetiva Object-Oriented.

Além disso, importa descrever os componentes não definidos pelo padrão e que se encontram representados no lado direito do componente da Figura 29:

- Authorization, é um componente responsável pelo cálculo de permissões de acesso a determinada informação por um ator (num determinado contexto). Mais detalhes sobre esta responsabilidade podem ser encontrados nas secções 5.4.3.3 e 5.4.4;

- Request, é um conjunto de classes auxiliares de gestão de pedidos REST;

- DTO, é um conjunto de classes que representam a informação de negócio de forma relevante no contexto do pedido, e.g. agregando objetos e escondendo particularidades da informação de domínio/modelo que não é vantajosa ou é mesmo indesejado que sejam conhecidas nesse contexto;

- Database, é um sistema de gestão de base de dados relacional responsável pela persistência de dados, servindo os pedidos de pesquisa e comando do Repository.

Apesar dos processos de negócio serem detalhadamente diferentes consoante a UC, são concetualmente semelhantes, o que permite a descrição das interligações entre os vários componentes de SWiPE BE. Para a sua descrição recorre-se a dois processos:

- Criar proposta de projeto (Figura 30), iniciada por um pedido REST, recebido pelo Controller que executa uma verificação rápida e superficial de permissões de execução do pedido através de Authorization. De seguida, delega a sua execução no Service, que verifica as permissões de execução da função e de acesso à informação com mais detalhe e contextualmente (cf. secção 4.2.1). Em caso de permissões, cria a proposta de projeto via Repository, e este delega na base de dados a persistência dos dados. A resposta é um DTO em que se devolve informação sobre o recurso criado (proposta de projeto) e ações que o ator pode realizar sobre ele. Para isso, usa-se o conceito de HATEOAS;

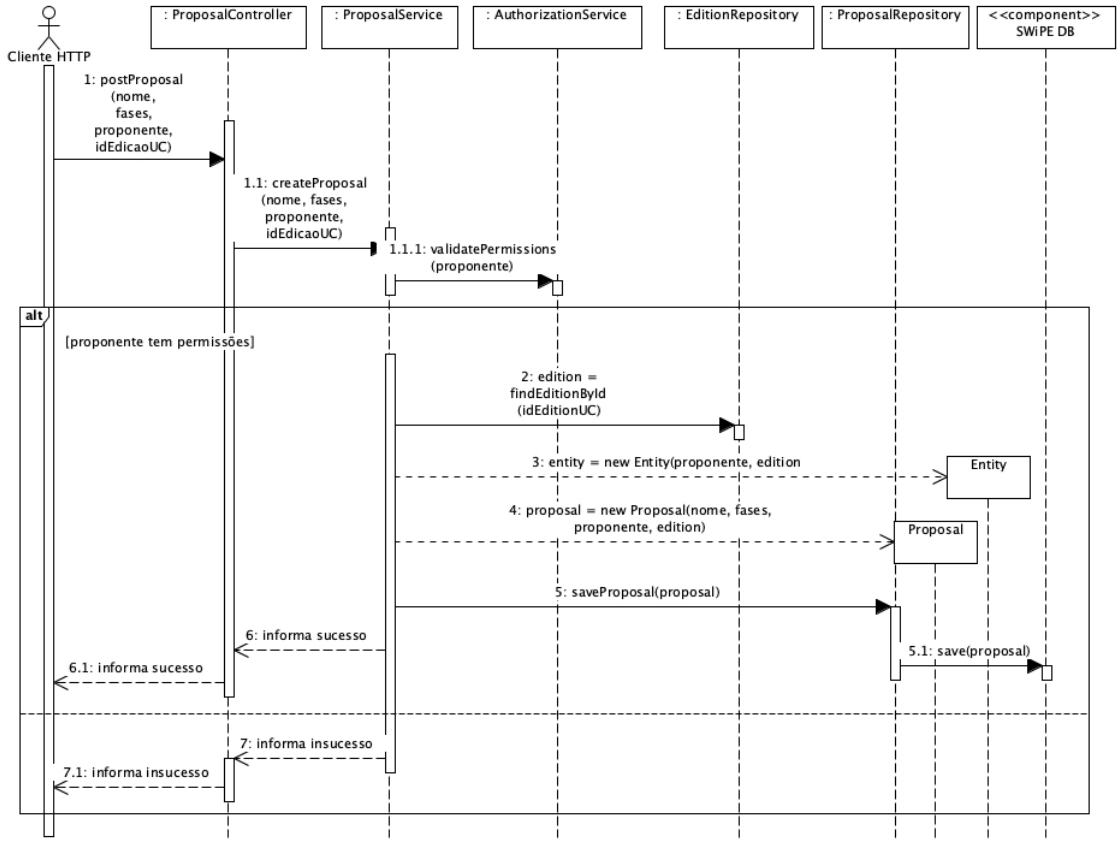

Figura 30 – Diagrama de sequência – Criar proposta no SWiPE BE

- Obter lista de anos letivos (Figura 31), iniciada por um pedido REST, recebido pelo Controller que executa uma conversão do token recebido no email do utilizador autenticado. De seguida, é delegado ao Service a obtenção da referida lista, que verifica as permissões do utilizador recorrendo ao Authorization. Caso, o utilizador tenha permissões, é obtida a lista (Repository) e são construídos os links que o utilizador pode executar em cada um dos anos letivos (HATEOAS), após este processo é retornada a lista de anos letivos conjuntamente com os links de cada ano letivo.

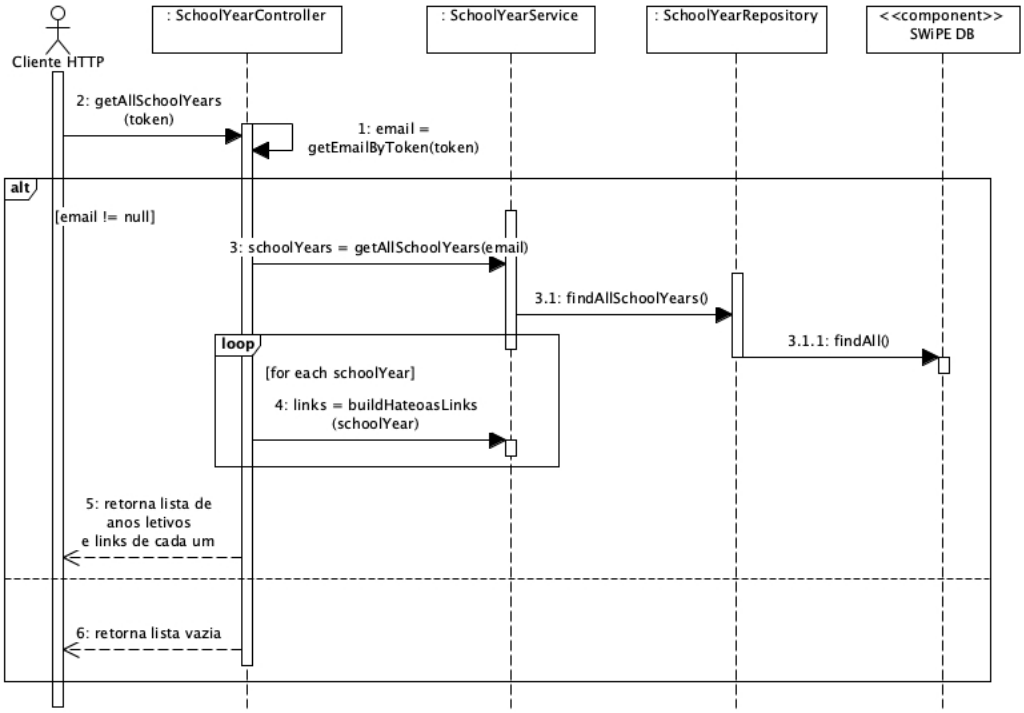

Figura 31 – Diagrama de sequência – Obter lista de anos letivos no SWiPE BE

Esta descrição não evidencia duas funcionalidades fundamentais do sistema: Configuração e Autorização. Por essa razão, estes dois requisitos-responsabilidades-funcionalidades serão descritos nas secções 5.4.3 e 5.4.4, respetivamente.

5.4.2 **Vista de módulos (Use)**

A Figura 32 representa os módulos de implementação do SWiPE BE, cada um correspondendo a um componente na vista C&C.

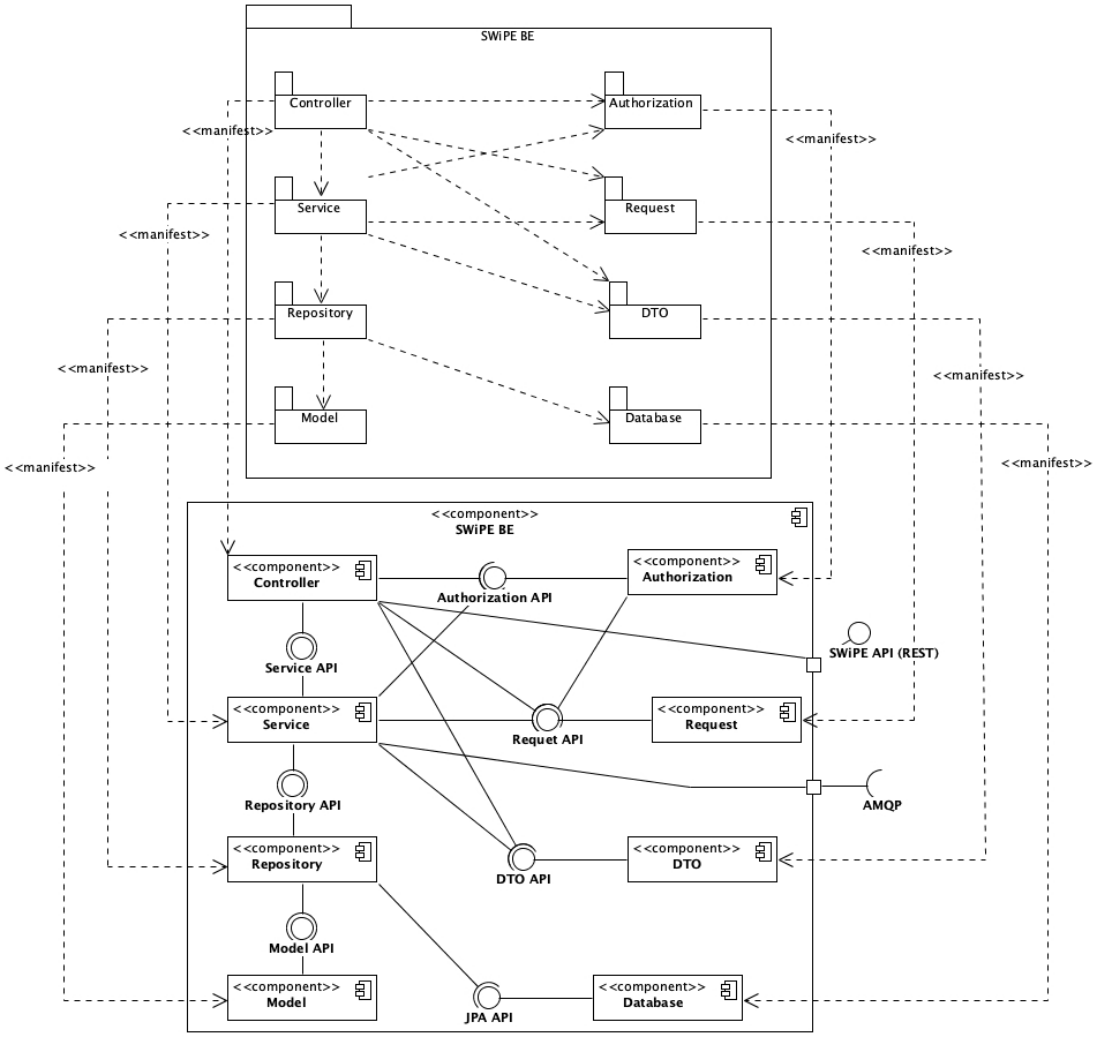

Figura 32 – Vista de módulos – SWiPE Backend

5.4.3 **Configurabilidade**

O requisito de configurabilidade é de fundamental importância neste projeto, já que permitirá que cada edição de UC tenha o funcionamento desejado.

A informação resultante das elicitação foi sistematizada em três dimensões complementares:

- Informação, como o conjunto organizado de dados necessário para a gestão da UC e de cada projeto/estágio. Qualquer projeto/estágio tem sempre duas partes distintas de informação: proposta e projeto, sendo que algumas propostas dão origem a projetos. Propostas e projetos passam por fases/estados distintos em função da edição da UC;

- Processos, como as funções/ações de introdução e transformação de informação executadas na gestão de (edição de) UC (e.g. (i) RUC introduz júri da prova, (ii) docente aceita arguir a prova);

- Atores, como as pessoas que executam os processos de gestão da UC de acordo com os papéis (e.g. RUC, orientador, estudante, arguente, proponente) e respetivas permissões contextuais (e.g. um docente tem acesso ao relatório se e só se for orientador, arguente ou presidente do júri da prova). As permissões de acesso a informação e execução de processos variam por (edição de) UC.

5.4.3.1 Modelo de classes

Porque é necessário que o espaço combinatório destas dimensões seja definido especificamente para cada (edição de) UC refletindo as suas especificidades para melhor suporte à gestão, uma das partes nucleares do sistema é a capacidade de, para cada (edição de) UC, representar/capturar dois momentos de atuação:

- Configuração, correspondendo à captura/descrição por parte do configurador, das particularidades de informação, processos e intervenientes de cada edição de UC;

- Operação, correspondendo ao funcionamento da edição da UC, de acordo com a configuração por partes dos intervenientes (e.g. RUC, orientador, estudante, proponente, arguente).

Para capturar estes dois momentos, há que distinguir entre classes de configuração e classes de operação (Figura 33).

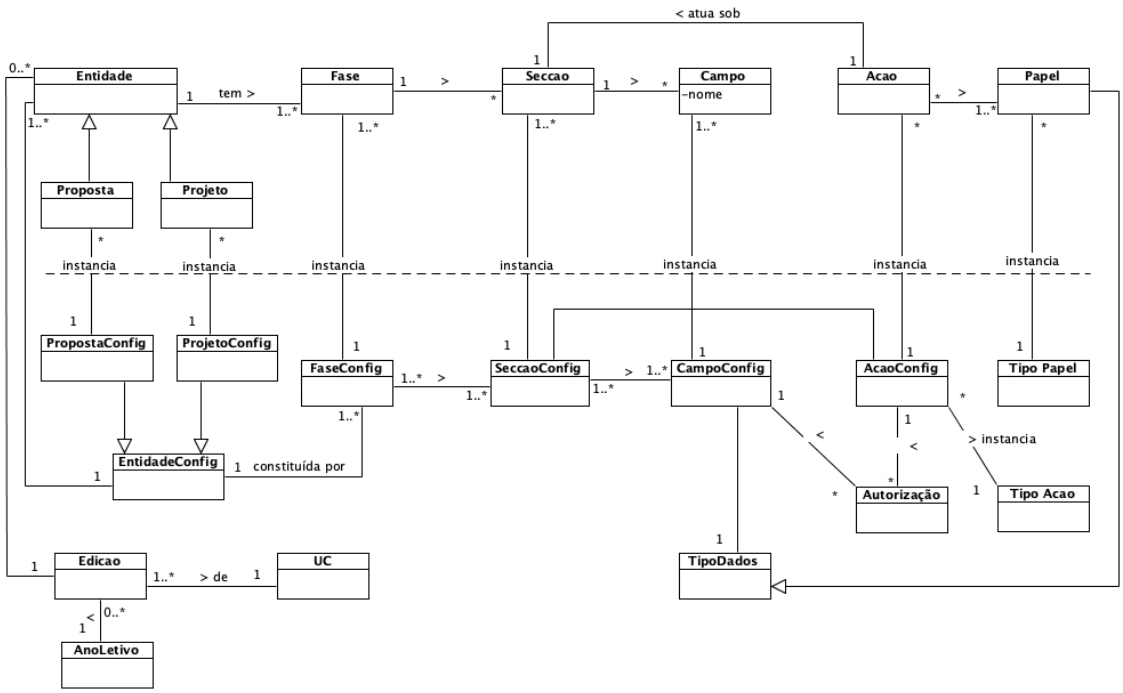

Figura 33 – Vista geral de classes de Model

Para capturar estes dois momentos, há que distinguir entre classes de configuração e classes de operação (Figura 33).

Abaixo da linha tracejada representam-se as classes de configuração, e acima, as classes de operação; i.e. as classes que instanciam as classes de configuração.

O ponto inicial lógico do modelo de classes é, portanto, EntidadeConfig, que representa o entendimento que cada UC faz dos conceitos de Proposta ou Projeto, pelo que são especializadas em PropostaConfig e ProjetoConfig.

Qualquer uma destas classes contém uma lista de FaseConfig, cada uma contendo uma lista de SecçãoConfig, e cada uma destas uma lista de CampoConfig. FaseConfig representa subdivisões lógicas da informação e do processo, capaz de representar diversos eventos letivos ao longo da proposta ou projeto (e.g. uma submissão parcial ou final). Por outro lado, cada SecçãoConfig é a especificação da informação (representada pela estrutura que se inicia em CampoConfig) que deve ser manipulada/transferida/persistida por cada ação (AcçãoConfig) dos atores ao longo do ciclo de vida da entidade (i.e. proposta ou projeto) conforme a Autorização definida.

No contexto de operação, Proposta será instanciada com propostas elaboradas por organizações e docentes (e outros atores definidos) conforme a estruturada em fases e secções definida em PropostaConfig. O mesmo raciocínio se aplica a ProjetoConfig e à sua contraparte de operação: Projeto.

Enquanto PropostaConfig e ProjetoConfig são classes criadas e geridas em tempo de configuração por Administrador, em tempo de operação Proposta e Projeto são manipuladas (criadas e geridas) pelos outros atores (cf. secção seguinte).

5.4.3.2 Detalhes do modelo de classes

Alguns conceitos do negócio pressupõem uma semântica mais rica do que a representada e que terá de ser capturada pelo código pois seria pouco útil uma representação visual de todas as restrições aplicáveis.

Contudo, os diagramas das 3 figuras seguintes complementam um pouco a semântica anterior (Figura 34), com detalhes sobre:

- Tipo Campo, especializado em Número, Texto, Data-Hora, Estado, Local, Booleano, Documento, Estrutura (conjunto de campos), Lista (conjunto de instâncias do mesmo tipo de campo), ... e Tipo Papel (Figura 34);

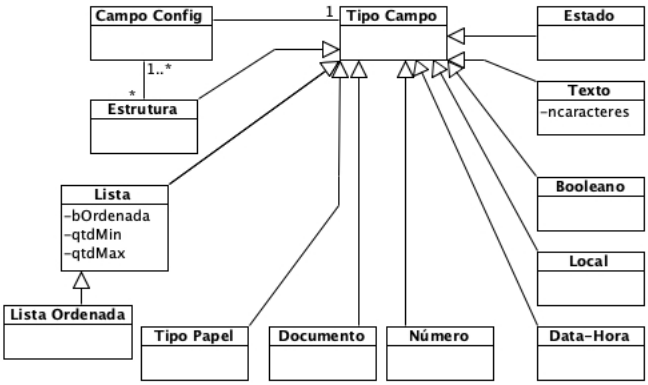

Figura 34 – Detalhe do diagrama de classes sobre a classe Tipo de Campo

- Tipo Papel, captura os papéis que os atores desempenham no negócio e é especializado em três hierarquias (Figura 35):
    - Tipo Papel Negócio, correspondente a papéis estáticos do ator;
    - Tipo Papel em Contexto, que captura os pepéis que determinado ator pode desempenhar num contexto de informação (e.g. proposta de projeto, projeto);
    - Combinação das duas anteriores, em que qualquer destas subclasses é subclasse de uma subclasse de Tipo Papel Negócio e subclasse de Tipo Papel Contexto.

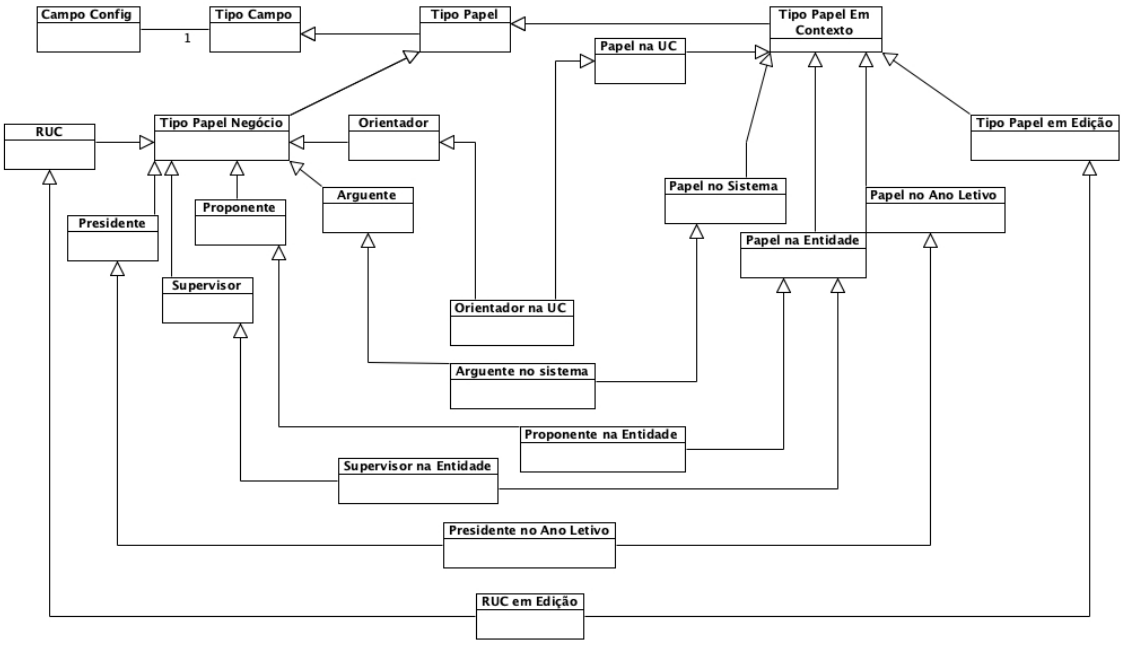

Figura 35 – Detalhe do diagrama de classes sobre a classe Tipo de Papel

Complementarmente, identificaram-se 4 tipos de ações configuráveis (Figura 36):

- Leitura de informação/campos de proposta ou projeto;
- Criação de propostas ou projetos, que ocorrem no momento da criação duma proposta de projeto ou dum projeto. Esta operação corresponde a um pedido com o método POST com os valores dos campos definidos na primeira secção da primeira fase da entidade;
- Conversão de projetos em propostas, que pode ocorrem em alternativa à criação de projeto (cf. ponto anterior). Esta operação corresponde a um pedido com o método POST em que os valores de criação de projeto são definidos por configuração a partir da informação/campos existentes na proposta de projeto;
- Alteração de informação de proposta ou projeto, e que manipula a informação/campos de uma secção de uma fase;
- Envio de notificações a atores/destinatários específicos, e cujo conteúdo da notificação advém da informação da secção em que ocorre.

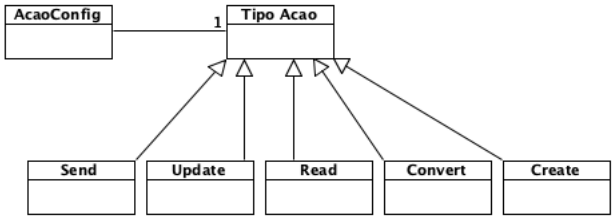

Figura 36 – Detalhe do diagrama de classes sobre a classe Tipo de Ação

5.4.3.3 Ação e autorização

Aquando da configuração, é o configurador/administrador que define o tipo de ação a realizar de cada secção de cada fase de cada entidade. Portanto, corresponde à existência de uma ação de um dos tipos anteriores. As fases e secções são configuradas definindo a ordem por omissão da sua realização. Sempre que a ação é realizada sobre uma secção, o estado da entidade (i.e. projeto ou proposta) passa a refletir a execução dessa ação. Portanto, o estado da entidade é conforme a última ação realizada sobre a entidade.

A ação de cada secção é realizável por um ou mais atores, consoante as autorizações definidas. Contudo, as autorizações não são definidas na granularidade de secção, mas na granularidade de Campo (definido em secção), i.e. instância de Autorização correlacionando Campo Config, Tipo Ação e Tipo Papel.

Uma das formas de representar esta informação é através duma tabela, como a exemplificada na Tabela 8 seguinte:

Tabela 8 – Exemplo de informação de autorização

CampoConfig	Papéis																		
	Estáticos				Dinâmicos														
					em Fase					em Entidade					em Sistema				
	Administrador	Docente	Estudante	Externo	RUC	Orientador	Coorientador	Supervisor	...	RUC	Estagiário	Orientador	Coorientador	...	RUC	Estagiário	Orientador	Coorientador	Supervisor	...
Campo 1	*	R	R	R	*	R	R			*	R	R	R		R					
Campo 2	*				*	U	R			*	U	R	R		R					
Campo 3	*				*	R				*		R								
Campo 4	*				Cv	R	R			Cv	R	U	R							
...																				

Legenda: *: Permissões completas; R: Permissão de Read (Leitura); C: Permissão de Create; (Criação); U: Permissão de Update (Alteração); N: Permissão de Notificação; Cv: Permissão de Conversão.

Esta informação de autorização é, portanto, configurável por (edição de) UC e utilizada no processo de autorização, descrito na secção 5.4.4.

5.4.3.4 API REST

Considerando a distinção entre funcionalidades de configuração e operação, e seguindo o princípio SOLID Interface Segregation Principle (SRP), a SWiPE API REST deve ser segregada em duas interfaces (Figura 37):

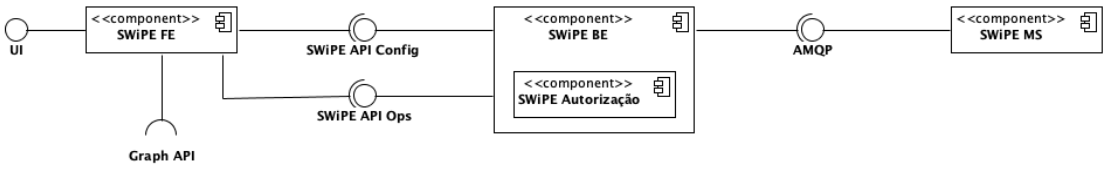

Figura 37 – Segregação da SWiPE API REST

- SWiPE API Config, relativa aos métodos de configuração, e.g.:
    - changeProposalConfigByEditionSchoolYear (`/schoolyear/{schoolYearId}/edition/{editionId}/proposalconfig/{proposalConfigId}`) – permite alterar as configurações definidas anteriormente
    - getPhasesConfigByProposalConfigEditionSchoolYear (`/schoolyear/{schoolYearId}/edition/{editionId}/proposalconfig/{proposalConfigId}/phaseconfig`) – obter a lista de todas as fases config que constituem uma proposta config

- createFieldConfigBySectionConfigPhaseConfigProposalConfigEditionSchoolYear (`/schoolyear/{schoolYearId}/edition/{editionId}/proposalconfig/{proposalConfigId}/phaseconfig/{phaseConfigId}/sectionconfig/{sectionConfigId}/fieldconfig`) – criar um campo config num determinado context (secção config, fase config, proposta config, edição, ano letivo)

- SWiPE API Ops, relativa aos métodos de operação, e.g.:
  - getAllSchoolYears (`/schoolYear`) – obter a lista de todos os anos letivos configurados no sistema (Figura 31)
  - createProposal (`/schoolYear/{id}/edition/{id}/proposal`) – criar proposta num determinado ano letivo e edição (Figura 30)
  - welcome (`/welcome`) – obter links da página inicial (Figura 24)

5.4.4 **Autorização**

O componente Autorização mencionado em 5.3.1.3 é responsável por calcular/determinar a permissões de um utilizador realizar uma ação sobre uma informação.

Contudo, o processo de controlo de permissões é mais do que o cálculo de permissões, e exige a participação de outras partes do sistema.

O XACML (eXtensible Access Control Markup Language) (OASIS, 2001) é um conjunto de recomendações relacionadas com o controlo de acesso a recursos. Em particular para este projeto, torna-se relevante porque propõe uma arquitetura e um modelo de processamento de da informação para controlar o acesso aos recursos.

Essa arquitetura sugere 4 pontos, correspondentes na nomenclatura usada neste documento como componentes:

- *Policy Administrator Point* (PAP);
- *Policy Enforcement Point* (PEP);
- *Policy Information Point* (PIP);
- *Policy Decision Point* (PDP).

5.4.4.1 Policy Administrador Point

PAP é o componente do sistema responsável por criar e gerir as regras de autenticação (WSO2, 2019). No SWiPE, o PAP é o componente de Autorização, dado que é este que carrega as configurações das regras definidas pelo administrador do sistema.

5.4.4.2 Policy Enforcement Point

PEP é o componente do sistema que força a aplicação das permissões do utilizador, garantindo assim o cumprimento das autorizações definidas. Portanto, este componente solicita a decisão de autorização ao PDP (WSO2, 2019).

Neste sistema, existem vários componentes PEP, nomeadamente:

- SWiPE FE: este componente apresenta as informações ao utilizador, mediante a informação transmitida pelo SWiPE BE. Desta forma, os dados são apresentados mediante as decisões tomadas pelo PDP;

- Componentes Controller e Service do SWiPE BE, na medida em que, antes do processamento do pedido, solicitam a decisão de permissão ao PDP.

5.4.4.3 Policy Information Point

PIP é o componente responsável por armazenar a informação necessária para a tomada de decisões sobre autorização (WSO2, 2019).

No sistema apresentado o PIP é a base de dados, visto que é nela que se encontram armazenados os dados necessários para o calcula de autorização, seja a informação de negócio.

5.4.4.4 Policy Decision Point

PDP é o componente responsável por avaliar as condições de acesso aos dados. É este componente que decide se um utilizador tem ou não acesso a determinada informação (WSO2, 2019).

No sistema, o PDP é o componente de autorização do SWiPE BE, e neste o Authorization. Este componente é responsável por reunir a informação necessária à decisão a partir do PIP, e assim decidir sobre a permissão de acesso ou não. As informações apresentadas no sistema, estão dependentes das decisões tomadas pelo PDP.

A arquitetura descrita em cima, é representada nos diagramas da Figura 38 e da Figura 39.

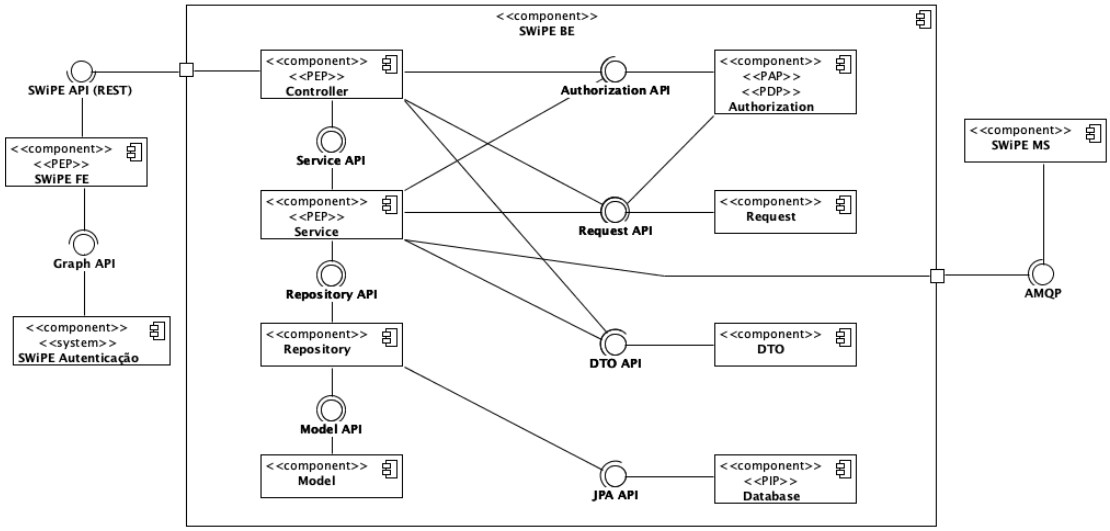

Figura 38 – Arquitetura segundo XACML

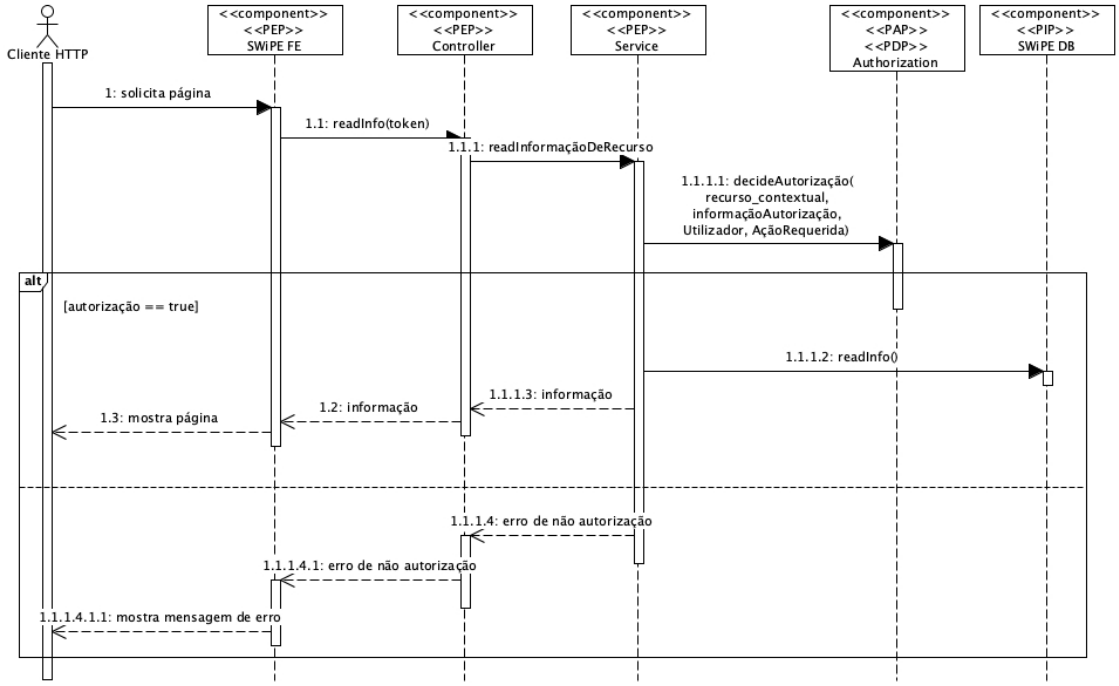

Figura 39 – Diagrama de sequência segundo XACML

5.5 Design arquitetural de aplicação SWiPE Frontend (Nível 3)

O objetivo é desta secção é descrever a arquitetura dum protótipo que sirva de prova de conceito do próprio componente SWiPE FE, mas especialmente das funcionalidades do SWiPE BE.

O SWiPE FE é uma *Rich Internet Application* (RIA), dado que é executada num navegador web e não é necessária a sua instalação por parte do cliente (Rouse, 2007). Contudo, este componente é também uma *Single Page Application* (SPA), uma vez que é carregada uma única página HTML e esta é atualizada dinamicamente conforme a interação do utilizador (Wasson, 2013).

5.5.1 **Vista C&C**

A arquitetura interna do SWiPE FE segue uma abordagem *Model-View-Controller* (MVC). Este tipo de abordagem define três divisões do software (Fowler, 2006) (Fowler, 2002) (Figura 40):

- Model, responsável pela manipulação dos dados de negócio
- View, responsável pela apresentação da informação aos utilizadores
- Controller, responsável pela gestão da informação a apresentar

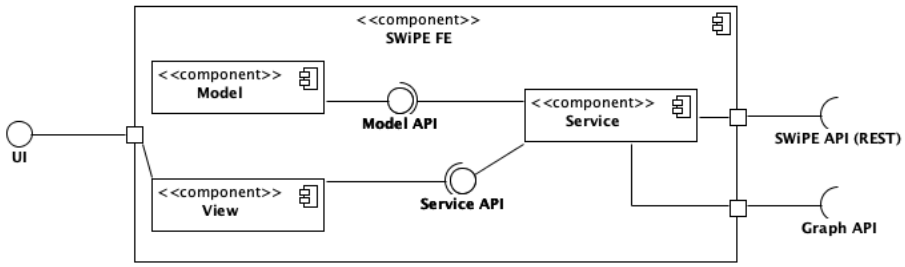

Figura 40 – Vista C&C – SWiPE FE

5.5.2 **Vista comportamental**

A Figura 41 representa o funcionamento do SWiPE FE quando um utilizador (administrador) pretende consultar a lista de anos letivos.

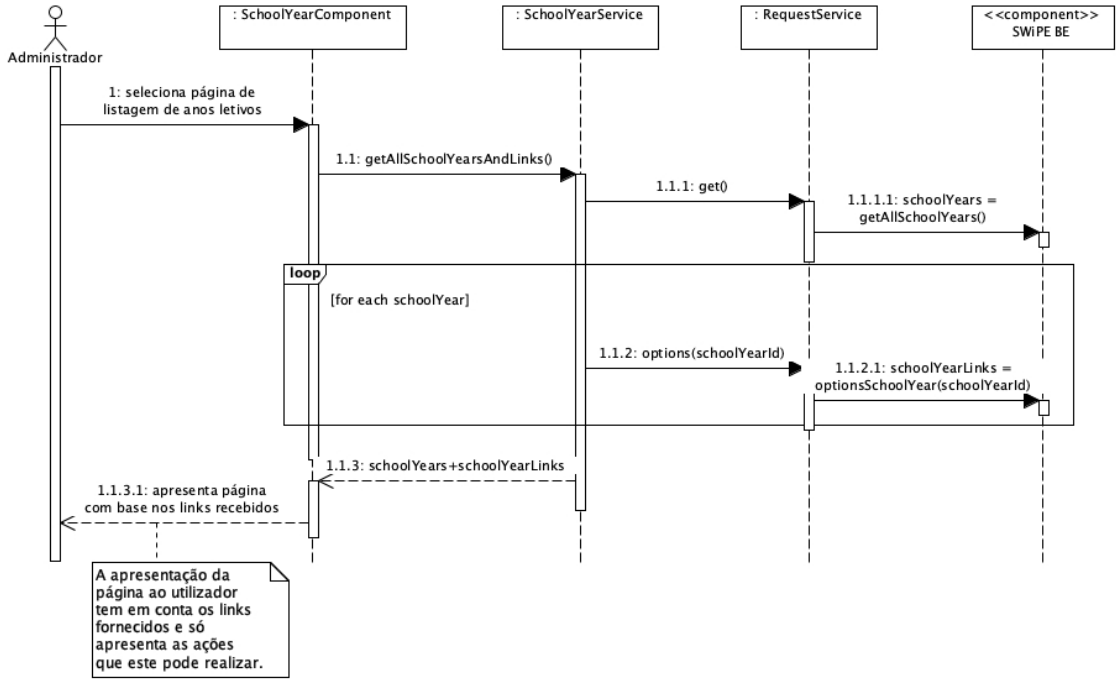

Figura 41 – Consultar lista de anos letivos (notação UML)

A Figura 42 representa o funcionamento do SWiPE FE quando um utilizador (proponente) pretende criar uma proposta.

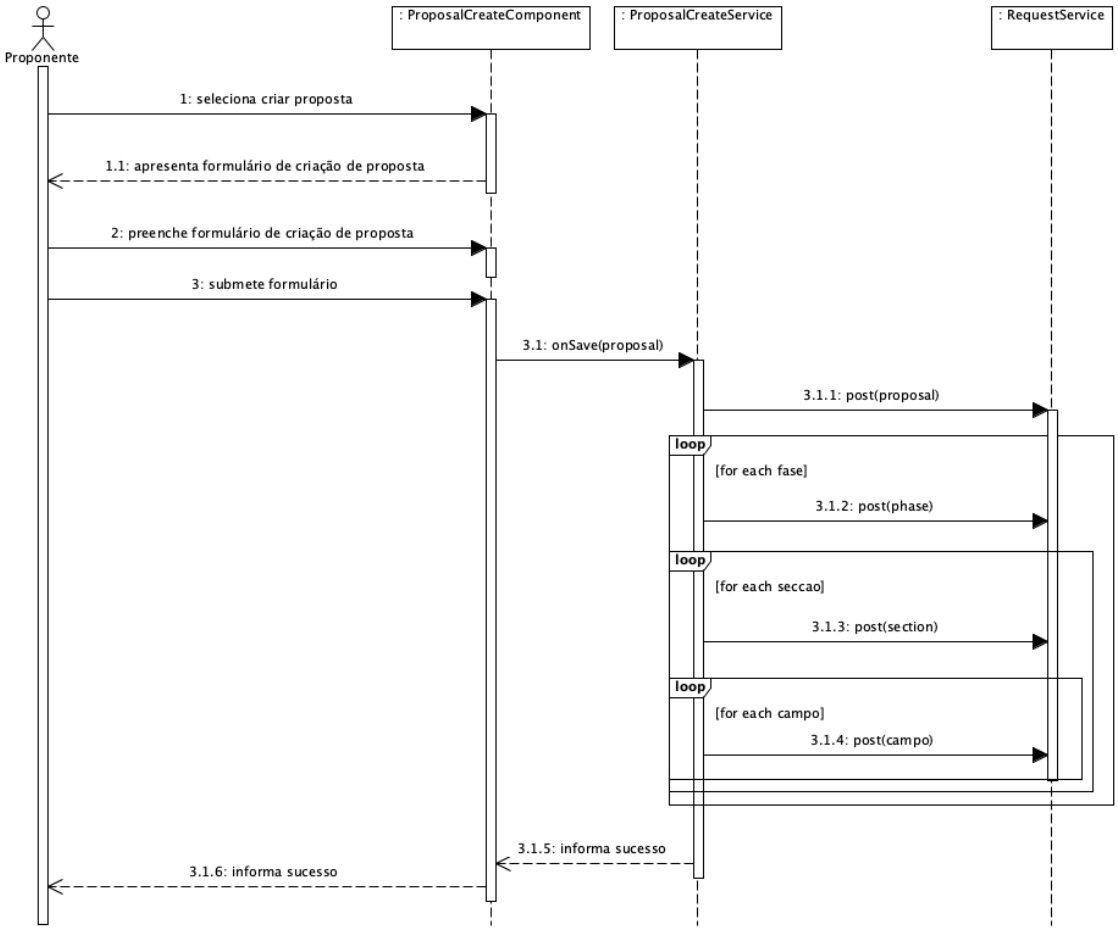

Figura 42 – Criar proposta SWiPE FE (notação UML)

5.5.3 **Vista de módulos**

A Figura 43 representa os módulos de implementação do SWiPE FE, seguindo como já referido, uma abordagem MVC.

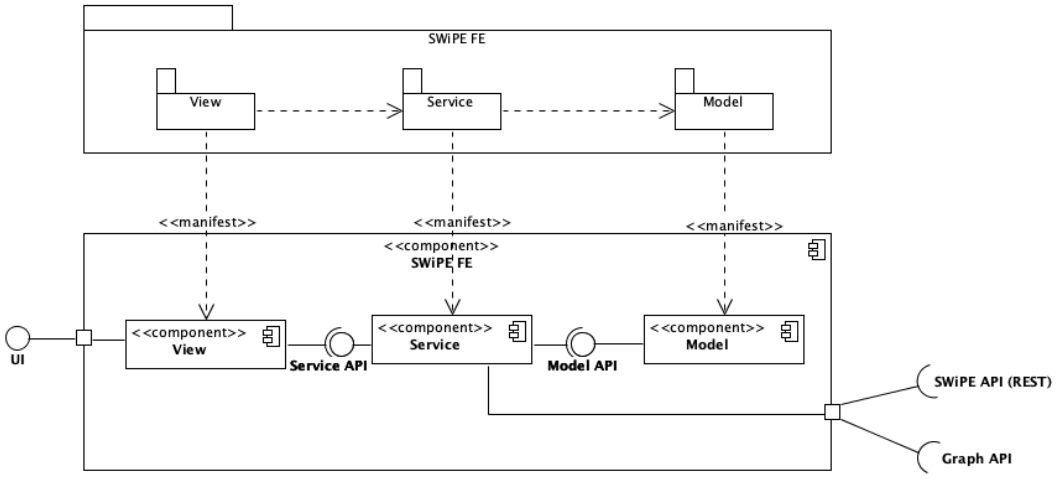

Figura 43 – Vista de módulos - SWiPE Frontend

5.6 Síntese

Este capítulo descreveu os elementos arquiteturais fundamentais do sistema de software preconizados.

Sempre que possível e tentando não tornar as explicações demasiado exaustivas, foram descritas as alternativas, racionais e decisões tomadas, apresentando sempre em UML as decisões tomadas.

6 Implementação

Este capítulo descreve algumas das principais tarefas de implementação dos componentes descritos no capítulo anterior, evidenciando algumas das principais decisões sobre tecnologia e sobre a própria implementação e implantação.

6.1 Decisões tecnológicas

As próximas secções têm como objetivo apresentar as decisões tecnológicas em cada um dos componentes que constituem o sistema.

6.1.1 **SWiPE BE**

Ao nível do SWiPE BE, para além de serem apresentadas as alternativas tecnológicas, é apresentada a implementação do mecanismo de autorização.

6.1.1.1 Framework e linguagem

No que diz respeito à escolha da linguagem de implementação, foram consideradas três linguagens/*frameworks* alternativos:

- Java Spring Boot (Software, 2019);
- ASP.NET Core (Rick-Anderson, 2019);
- NodeJS (N. js Foundation, 2019).

A Tabela 9 representa os critérios que foram tidos em conta e o cumprimento desses em cada uma das tecnologias.

Tabela 9 – Comparação entre tecnologias – SWiPE BE

	Java Spring Boot	**ASP.NET Core**	**NodeJS**
Programação orientada a aspetos (AOP)	Sim	Sim	Sim
Desenvolvimento de API REST	Sim	Sim	Sim
Tamanho da comunidade	Muito grande	Grande	Grande
Experiência pessoal	Sim	Sim	Não

Posto isto, a linguagem/tecnologia escolhida foi Java Spring Boot, uma vez que existia uma maior experiência na utilização desta por parte da autora

6.1.1.2 AspectJ

Para utilizar esta extensão foi definida uma anotação, que tem como objetivo identificar os métodos em que é necessário verificar as permissões do utilizador antes da sua execução. Esta anotação permite criar o contexto de informação a que se pretende aceder e, desta forma, garantir que o mecanismo de autorização possui toda a informação necessária para decidir.

Antes da execução dos métodos assinalados com a anotação mencionada, são verificadas as permissões do utilizador autenticado. Caso, este não tenha permissões para aceder à funcionalidade o método não é executado e essa informação é transmitida ao utilizador.

6.1.2 **SWiPE FE**

Ao nível do SWiPE FE são apresentadas a alternativas tecnológicas, bem como bibliotecas que foram utilizadas no desenvolvimento desta aplicação. Finalmente, são apresentadas algumas imagens do resultado final do protótipo desenvolvido.

deste documento. Porque se pretende que a aplicação cliente seja do tipo *Single Page Application* (SPA), deverá ser adotado *frameworks* ou bibliotecas que facilitem e potenciem o seu desenvolvimento. Para tal, analisaram-se três *frameworks*/bibliotecas:

- Angular (Google, 2010);
- React (Facebook Inc., 2013);
- Vue.js (You, 2014).

A Tabela 10 representa os critérios que foram tidos em consideração para a seleção da tecnologia a adotar.

Tabela 10 – Comparação entre tecnologias - Frontend

	Angular	**React**	**Vue.js**
Divisão em componentes	Sim	Sim	Sim
Desenvolvimento de SPA	Sim	Sim	Sim
Integração com API REST	Sim	Sim	Sim
Tamanho da comunidade	Grande	Grande	Médio
Instalação de bibliotecas de suporte	npm	npm	npm
Experiência pessoal	Sim	Não	Não

Assim, a linguagem escolhida foi o Angular, visto que existia uma maior experiência pessoal, mas também uma vontade de melhorar os conhecimentos nesta tecnologia.

6.1.2.1 RxJS

O RxJS (Lesh et al., 2019) é uma biblioteca de programação reativa, que utiliza *Observables* para a construção dos resultados dos pedidos. Esta biblioteca fornece uma interface que permite simplificar a utilização de *Observables*.

Desta forma, é possível realizar os pedidos HTTP ao SWiPE BE de forma assíncrona, mas garantindo que estes apenas são executados quando já está disponível toda a informação. Por isso, quando o SWiPE FE necessita de fazer pedidos com informações relacionadas, a utilização do RxJS permite garantir que os pedidos-filhos só são executados quando o pedido-pai terminou.

6.1.2.2 Interface com utilizador

As Figuras Figura 44 e Figura 45 representam exemplos de interfaces gráficas a que o utilizador final tem acesso no SWiPE FE.

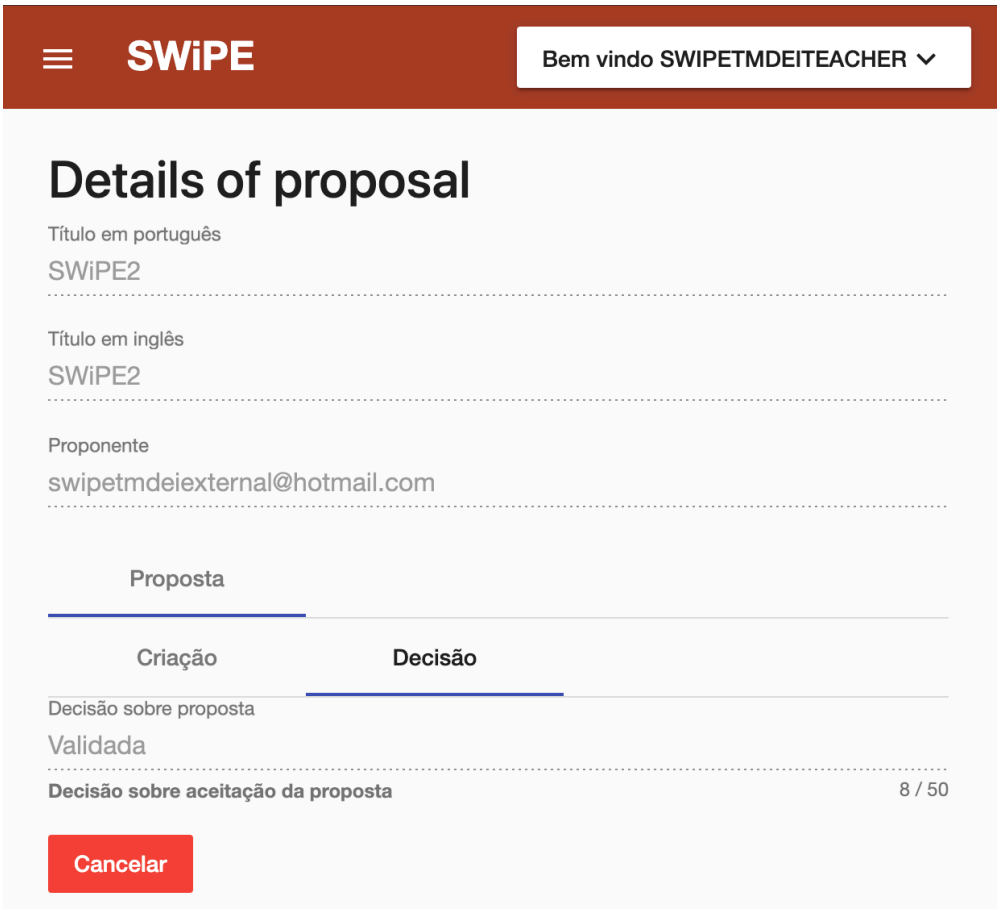

Figura 44 – Exemplo de ação sob proposta

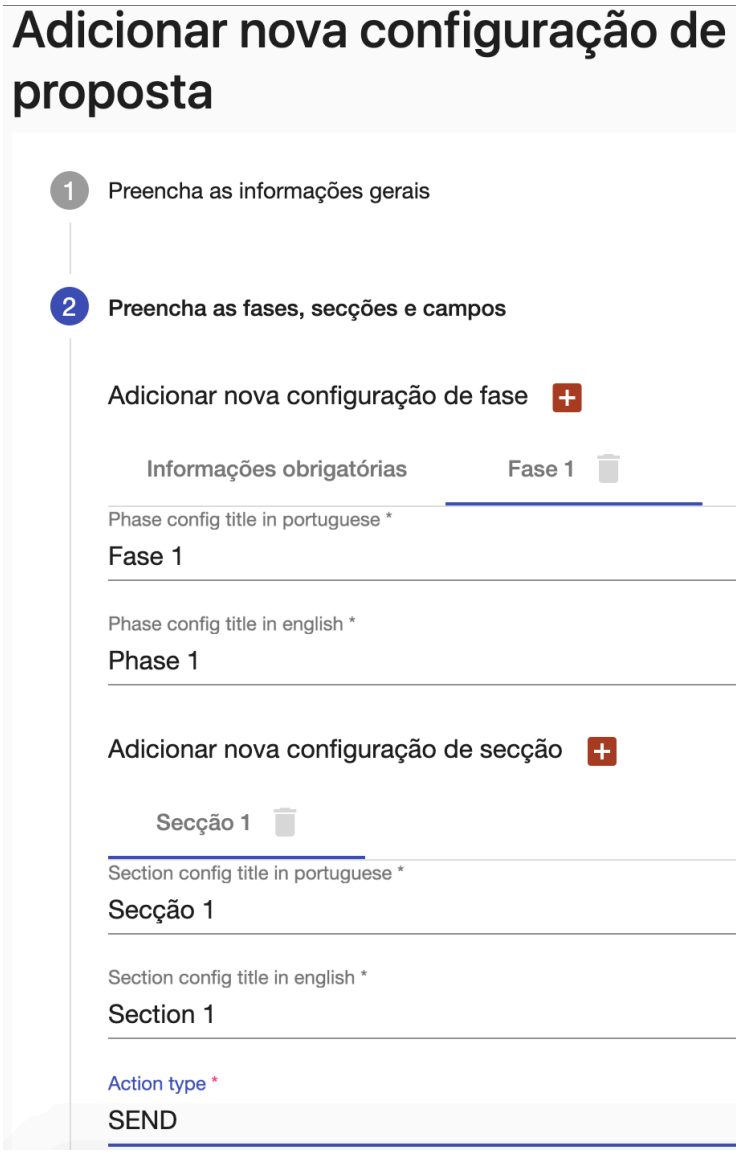

Figura 45 – Exemplo de página de criação de configuração de propostas

6.1.3 **SWiPE Autenticação**

Dado que, já havia sido decidido utilizar um sistema de autenticação externo (cf. Secção 5.3.1.2), foi necessário escolher qual o mecanismo a adotar. As contas de email institucionais do ISEP são registadas num servidor *Microsoft*, por isso a autenticação através deste servidor de email seria simplificada para a comunidade escolar. Por outro lado, também existia a necessidade de empresas/organizações externas acederem ao sistema, pelo que foi necessário estudar a possibilidade de outras contas utilizarem este mecanismo.

Após ser desenvolvido um protótipo foi possível concluir, que após o registo de contas não *Microsoft* na aplicação, era possível que estas se registassem no servidor *Microsoft*, permitindo assim o acesso externo.

Assim, foi decidido utilizar o *Azure Active Directory* (AAD) (Microsoft, 2019a), visto que, este serviço da *Microsoft* permite gerir acessos e identidades.

6.1.4 **SWiPE *Message Store***

Como referido na secção 5.3.1.4, foi utilizada uma *Message Store* para armazenar os eventos produzidos no sistema. Estes eventos representam as alterações realizadas a uma determinada entidade. Assim, quando é executada uma ação a *Message Store* é notificada.

Porque existem diversas tecnologias potenciais, analisaram-se três:

- Kafka (Apache, 2019a);
- RocketMQ (Apache, 2019b);
- RabbitMQ (Pivotal, 2019).

Apesar de todas estas tecnologias permitirem o envio de mensagens de forma assíncrona, existem fatores que permitiram escolher uma em detrimento das outras. A Tabela 11 permite sistematizar os critérios de decisão.

Tabela 11 – Comparação de tecnologias – *Message Store*

	Kafka	**RocketMQ**	**RabbitMQ**
Utilização de serviços complementares	Sim	Não	Não
Comunidade	Muita experiência	Pouca experiência	Muita experiência
Experiência pessoal	Não	Não	Sim

Posto isto, a tecnologia escolhida foi o RabbitMQ, dado que esta possui elevada qualidade na documentação oficial apresentada e já existia alguma experiência na sua utilização.

6.1.5 **Base de dados**

A base de dados do sistema podia ser de dois tipos:

- Base de dados relacional;
- Base de dados não relacional.

Uma base de dados relacional, como o próprio nome indica permite representar dados relacionados entre si. Para além disso, as bases de dados relacionais têm diversas regras de integridade, com vista a garantir que os dados armazenados são corretos. Neste tipo de base de dados, são criadas tabelas correspondentes a cada uma das entidades do domínio e cada uma dessas tabelas tem colunas correspondentes aos atributos das entidades. O modelo

relacional permite representar esta informação, bem como as relações existentes entre as tabelas (Oracle, 2019).

Por outro lado, uma base de dados não relacional não possui tabelas, utilizando por sua vez um sistema de armazenamento focado na informação a guardar. Este tipo de base de dados, centra-se sobretudo no armazenamento de documentos e não na estrutura genérica destes. Nestas bases de dados não é utilizado um modelo relacional (Microsoft, 2019c).

Uma vez que, as entidades deste sistema possuíam relações entre si, era necessário que o tipo de base de dados escolhida permitisse representar estas ligações e desta forma, garantir que o armazenamento destes dados era seguro e segundo um determinado esquema.

6.2 Ambiente de implementação

Nas secções seguintes será descrito o ambiente de implementação utilizado ao longo do desenvolvimento do projeto.

6.2.1 **Continuous Integration/Continuous Delivery**

Como descrito na Secção 3.4.7.1 é requerido a adoção de CI/CD, que permita automatizar os processos de compilação, teste e entrega final do software.

De forma a cumprir o requisito de adoção de CI/CD, decidiu-se adotar a plataforma Bitbucket *(Atlassian, 2019)*. Esta plataforma tem caraterísticas relevantes para esta questão, encontradas também noutras plataformas semelhantes (e.g. Github (Preston-Werner et al., 2008)), mas que estão disponíveis gratuitamente para os estudantes/comunidade do ISEP:

- Repositório (de código) com Controlo de Versões, onde é armazenado o código dos projetos de software;
- Pipeline de CI/CD, Bitbucket pipeline (Atlassian, 2019), que é executado sempre que existe um novo *push* para o repositório do controlo de versões.

O pipeline desenvolvido (Figura 46) foi dividido em dois passos (*steps*), sendo que em cada um deles são realizados os processos de:

- Compilação do código do projeto, por forma a garantir que não há erros sintáticos nem semânticos da linguagem;
- Testes, definidos no próprio projeto, sejam unitários ou de integração;
- Implantação (entrega) do software na máquina estipulada.

```yaml
image: maven:3.3.9
clone:
  depth: full
pipelines:
    default:
        - step:
            name: Clean, Build and Test swipe-backend
            services:
                - broker
            script:
                - cd swipe-backend
                - mvn -B clean install -U
                - mvn -B clean checkstyle::check checkstyle:checkstyle
                javadoc:aggregate package
                - scp -i ~/.ssh/id_rsa.pub swipe-backend-webservice/target/swipe.war
                nps@193.136.62.205:/opt/tomcat9/webapps
        - step:
            name: Build, Test and Deploy SwipeFE
            image: node:10.15.0
            script:
                - cd SwipeFE
                - npm install
                - npm run gitbuild
                - mkdir SwipeFE
                - cp -a dist/. SwipeFE/
                - npm run compodoc-coverage
                - npm run compodoc
                - scp -r -i ~/.ssh/id_rsa.pub SwipeFE/ nps@193.136.62.205:/opt/tomcat9/webapps
definitions:
  services:
    broker:
      image: rabbitmq:3
      ports:
        - 5672:5672
```

Figura 46 – Especificação do pipeline desenvolvido

Com este pipeline, o processo de integração e de entrega é espoletado por um simples *push* da ferramento de Git para o repositório com nenhuma intervenção do engenheiro de software.

6.2.2 **Ferramentas de análise**

Nas secções seguintes são descritas as ferramentas de análise de código utilizadas neste projeto. Estas permitem garantir uma validação na estruturação do código, com o objetivo de uma boa leitura do mesmo, facilitando assim a manutenibilidade do código.

6.2.2.1 Checkstyle

Checkstyle é uma ferramenta de análise de código (Burn, 2001), utilizada em aplicações Java. A utilização desta ferramenta neste projeto permitiu garantir que o código desenvolvido seguia algumas boas práticas de estruturação de código e desenvolvimento de Javadoc. Para utilizar esta ferramenta deve ser criado um ficheiro checkstyle.xml, onde são utilizados diversos módulos, que permitem fazer uso de diversos tipos de regras. A Figura 39 representa o ficheiro

checkstyle.xml utilizado no SWiPE BE. Este ficheiro inclui sobretudo módulos de validação do Javadoc e de estruturação do código (e.g. nome dos métodos, *imports* não utilizados).

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE module PUBLIC "-//Checkstyle//DTD Checkstyle Configuration 1.3//EN"
"https://checkstyle.org/dtds/configuration_1_3.dtd">
<module name="Checker">
    <module name="TreeWalker">
        <module name="ArrayTypeStyle" />
        <module name="DefaultComesLast" />
        <module name="FinalClass" />
        <module name="IllegalImport" />
        <module name="JavadocMethod">
            <property name="allowMissingParamTags" value="true" />
            <property name="allowMissingThrowsTags" value="true" />
            <property name="allowMissingReturnTag" value="true" />
            <property name="allowedAnnotations" value="Override, Test" />
        </module>
        <module name="MethodName" />
        <module name="MissingSwitchDefault" />
        <module name="PackageName">
            <property name="format" value="^[a-z]+(\.[a-z][a-z0-9]*)*$" />
        </module>
        <module name="SummaryJavadoc">
            <property name="forbiddenSummaryFragments"
            value="^@return the *|^This method returns |^A [{]@code [a-zA-Z0-9]+[}]( is a )" />
        </module>
        <module name="UnusedImports" />
    </module>
</module>
```

Figura 47 – Ficheiro checkstyle utilizado

6.2.2.2 TSlint

À semelhança do Checkstyle, o TSLint (Crockford, 2002) é uma ferramenta de análise de código, sendo esta utilizada em aplicações Javascript. Neste projeto, fez-se uso da configuração do TSLint utilizada pela plataforma Airbnb (Chesky et al., 2008), visto que este inclui toda a configuração desejada para este projeto, nomeadamente estruturação do código, de forma a facilitar a leitura do mesmo. A utilização desta pressupõe a criação de um ficheiro tslint.json, constituído pelos módulos necessários para a configuração desejada. Visto que, neste projeto utiliza-se uma configuração já existente, foi apenas necessário importar esta configuração (Figura 45).

```
{
  "extends": "tslint-config-airbnb"
}
```

Figura 48 – Ficheiro tslint.json

6.3 Tarefas

As secções seguintes apresentam as diversas tarefas realizadas no sistema e as decisões tomadas para a implementação de cada uma delas.

6.3.1 **Autenticação**

Para permitir o acesso às diversas funcionalidades é necessário que o utilizador se autentique no sistema. Esta autenticação irá permitir obter algumas informações sobre o utilizador e consequentemente enviar estes dados para o mecanismo de autorização.

Como já referido o mecanismo de autenticação utilizado foi o AAD (Secção 6.2.6). Para utilizar este mecanismo, é necessário registar a aplicação no Portal *Azure* e definir que tipo de contas (pessoais e/ou organizacionais) podem tentar autenticar-se na aplicação. O administrador deve convidar os utilizadores envolvidos no processo de gestão (estudantes, docentes e empresas ou organizações externas), de forma a que estes tenham acesso ao sistema.

Posto isto, quando os utilizadores aceitarem o convite de acesso ao sistema e se autenticarem no SWiPE, quem valida a sua identidade é a *Microsoft* e não a aplicação desenvolvida. Do sucesso desta operação é resultante um *token*. Este código será posteriormente enviado para o SWiPE BE, para que este forneça as informações, às quais o utilizador tem acesso.

6.3.2 **Autorização**

Após a realização do processo de autenticação é obtido um *token* de acesso à aplicação. Este *token* será utilizado no SWiPE BE, com vista à atribuição de permissões ao utilizador.

O mecanismo de autorização implementado no SWiPE BE tem por base AspectJ (E. Foundation, 2019). Esta extensão da linguagem Java permite de entre muitas funcionalidades intersetar a chamada de métodos e a realização de ações aquando desta. Por isso, quando um determinado utilizador realiza um pedido de obtenção de funcionalidades possíveis, o mecanismo de autorização antes de realizar o método correspondente, verifica se o *token* recebido tem permissões para realizar tal ação.

6.3.3 **Internacionalização e Localização**

Visto que, é possível existirem estudantes estrangeiros a frequentar as UC em que este sistema se insere, existe a necessidade de garantir que os utilizadores podem escolher o idioma em que pretendem que as páginas lhes sejam apresentadas. Por isso, foi necessário que o sistema estivesse preparado tanto para apresentar as informações em diferentes idiomas, como para armazenar esta informação.

6.3.3.1 Internacionalização e Localização no SWiPE FE

De forma, a permitir a apresentação das páginas em idiomas diferentes, neste componente foram criados ficheiros de internacionalização (i18n). Estes ficheiros possuem a tradução de todas as palavras e/ou frases apresentadas ao utilizador e a respetiva tradução para o idioma pretendido. Dado que, a proveniência dos estudantes é variada foi decidido prover três línguas:

- Português de Portugal;
- Inglês do Reino Unido;
- Inglês dos EUA.

A existência do idioma inglês em duas localizações distintas justifica-se pelo facto de permitir uma cobertura mundial mais vasta. Visto que o SWiPE foi desenvolvido em Portugal fazia todo o sentido que aplicação estivesse disponível no idioma mais falado neste país, o português.

Apesar de tudo, o sistema permite aos RUC realizarem configurações das suas UC, pelo que existia a necessidade de permitir que estes escolhessem os idiomas com que desejavam trabalhar. Por isso, a configuração dos campos de propostas e projetos permite ao utilizador selecionar os idiomas que este pretende utilizar. Assim, é possível que cada UC funcione com idiomas personalizados, sem que isso afete o funcionamento das demais.

6.3.3.2 Internacionalização e Localização no SWiPE BE

Contudo, o SWiPE BE também devia estar preparado para esta abordagem. Assim, por cada configuração de campo criada são associados todos os idiomas definidos pelo administrador. Quando uma entidade é requisitada, é necessário que seja enviado um cabeçalho correspondente ao idioma favorito do utilizador. Desta forma, a leitura da entidade será realizada de acordo com esta preferência. Caso a linguagem favorita do utilizador não se encontre definida, é utilizada a predefinição (português de Portugal).

6.3.4 **HATEOAS**

O HATEOAS (Fowler, 2010) é um componente de uma API REST. Este componente permite fornecer ao utilizador informação dinâmica sobre as ações que este pode realizar sob determinado recurso.

Neste sistema, o HATEOAS permite enviar ao SWiPE FE os links das ações que o utilizador autenticado pode realizar sobre determinado recurso, permitindo assim construir uma interface gráfica dinâmica e mediante as permissões dos utilizadores, sem que para tal seja necessária a existência de lógica de negócio adicional (Figura 49).

Figura 49 – Exemplo de resposta a pedido com HATEOAS

6.3.5 **Documentação**

A documentação neste projeto tem como principal objetivo a melhoria da manutenibilidade do código desenvolvido. Este facto permite que futuros desenvolvedores possam mais facilmente entender o código e dessa forma alterá-lo.

6.3.5.1 Javadoc

Foi elaborada documentação para cerca de 80% de código do SWiPE BE, usando Javadoc.

O Javadoc (Oracle, 2006) é uma ferramenta de documentação de código Java. Esta permite criar páginas HTML com todas as descrições de classes, métodos e variáveis (Figura 50 e Figura 51).

```
/**
 * Get the list of proposal config in edition and school year.
 *
 * @param request      request to respond with the list of proposal config
 * @param user         email address of the user who wants to get the list of
 *                     proposal config
 * @param schoolYearId school year id to search
 * @param editionId    edition id to search
 * @return http status of the response
 */
```

Figura 50 – Exemplo de um comentário em Javadoc

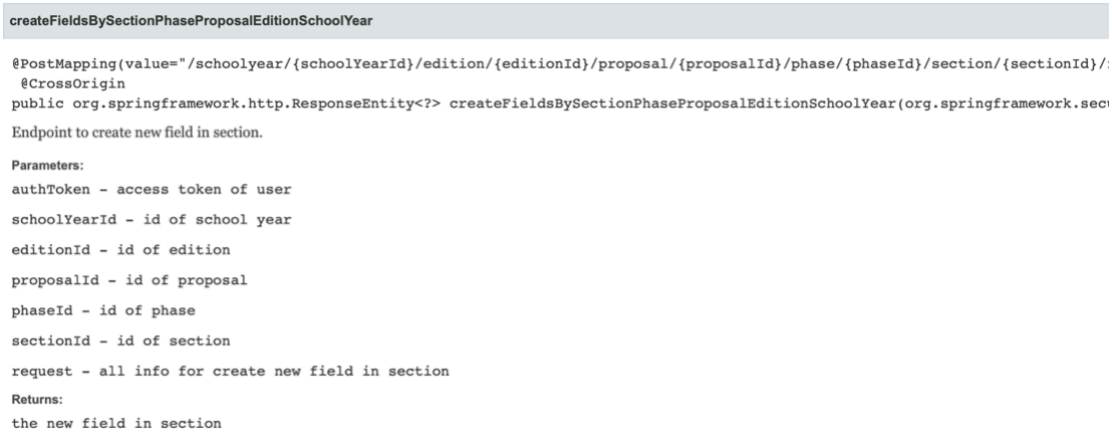

Figura 51 – Exemplo do Javadoc gerado a partir do comentário

6.3.5.2 Compodoc

Foi elaborada documentação para cerca de 70% de código do SWiPE FE, usando Compodoc.

À semelhança do Javadoc, o Compodoc permite criar documentação em aplicações Angular. Esta ferramenta permite criar validações de cobertura de documentação em cada ficheiro, garantindo assim, que toda a aplicação está devidamente documentada. O resultado da utilização do Compodoc é uma estrutura de página HTML com a informação sobre a aplicação (Figura 52 e Figura 53).

```
/**
* Get proposal config by id
*
* @param editionId Selected edition id.
* @param proposalConfigId Selected proposal config id.
*/
```

Figura 52 – Exemplo de um comentário Compodoc

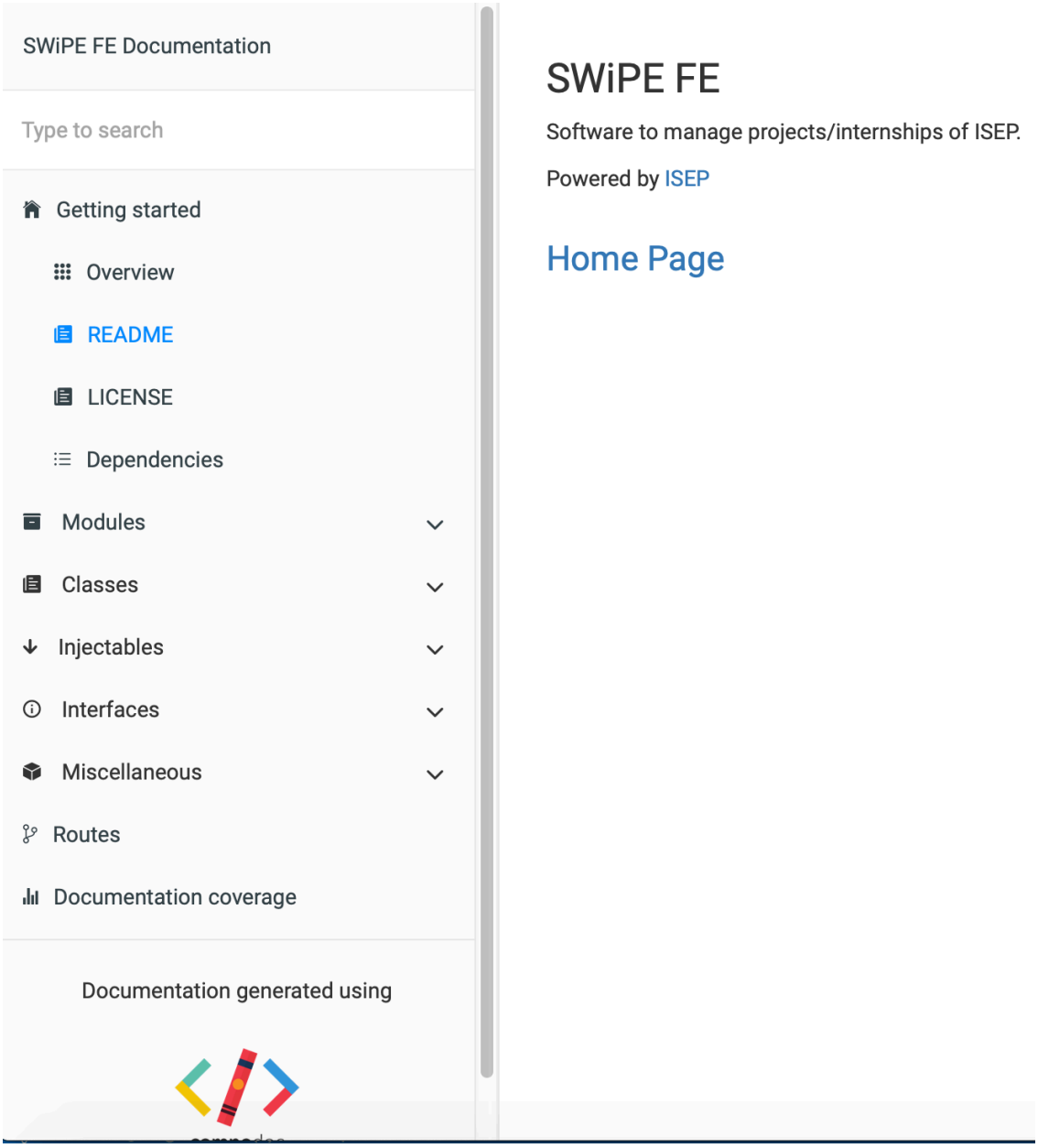

Figura 53 – Compodoc do SWiPE FE

6.4 Testes

Os testes de software permitem reduzir o risco de falhas durante a execução do software. Desta forma, antes de colocar um software em produção é necessário que este seja devidamente testado. De forma a facilitar este processo foram desenvolvidos testes automatizados, pois permitem reduzir o tempo gasto pelo programador na execução de testes manuais, mas também garantir que se o código base mudar, os testes desenvolvidos até então devem continuar a funcionar (Vocke, 2018).

Uma vez que a abordagem seguida no desenvolvimento deste projeto foi iterativa e incremental, foi utilizado o modelo *V-Model* (Figura 54). Este modelo pressupõe o desenvolvimento de testes em cada atividade de desenvolvimento do software (Firesmith, 2013).

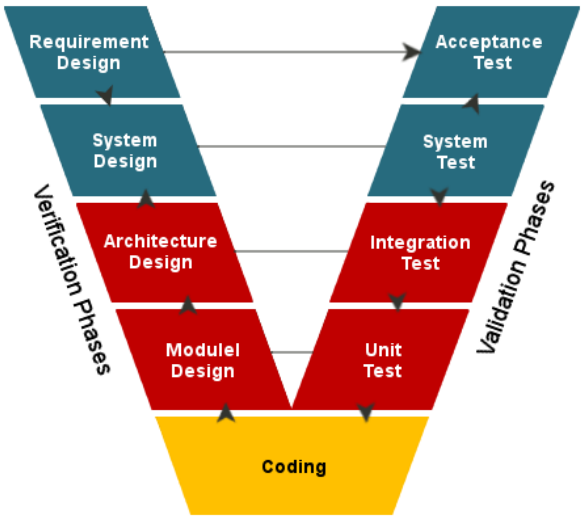

Figura 54 – Representação esquemática do *V-Model*[15]

No desenvolvimento deste projeto foram utilizados diferentes níveis de testes:

- Testes unitários;
- Testes de integração;
- Testes de sistema;
- Testes de aceitação.

6.4.1 **Testes unitários**

Os testes unitários focam-se em testar os componentes separadamente. Neste nível de testes as relações com outros componentes são simuladas para que seja possível testar o comportamento isolado do componente em causa. Para esta simulação são utilizados *stubs*[16] e *Drivers*[17]. Os testes unitários permitem sobretudo encontrar problemas em ciclos ou na lógica das funções desenvolvidas (ISTQB, 2017a).

Posto isto, foram desenvolvidos testes unitários no SWiPE BE e no SWiPE FE.

[15] Fonte: http://www.professionalqa.com/v-model
[16] Stub – Simula a chamada de outro componente por parte do componente alvo de teste.
[17] Driver – Simula a chamada do componente alvo de teste por parte de outro.

6.4.1.1 Testes unitários no SWiPE BE

Neste componente, os testes unitários focaram-se sobretudo na análise do mecanismo de autorização, de forma a verificar se este tem o comportamento esperado (Figura 55).

```
@Test
Run Test | Debug Test
public void isAuthenticatedUserTest() {
    Context ctx = new Context("student1@email.com");
    boolean result = authorizationService.isAuthenticatedUser(ctx);
    assertThat(result).isTrue();
}
```

Figura 55 – Exemplo de teste realizado ao mecanismo de autorização

Contudo, as funções de interação com a base de dados também são testadas, utilizando para tal, um *stub* da base de dados (Figura 56).

```
@Test
Run Test | Debug Test
public void whenFindByEmail_thenReturnUser() {
    User found = userRepository.findUserByEmail("email2@email.com");
    assertThat(found.getEmail()).isEqualTo("email2@email.com");
}
```

Figura 56 – Exemplo de teste realizado a uma função de interação com a base de dados

Os testes unitários do SWiPE BE foram desenvolvidos com recurso à *framework open source*: JUnit (Gamma and Beck, 2019).

6.4.1.2 Testes unitários no SWiPE FE

Ao nível deste componente foram desenvolvidos testes, na maioria dos casos, aos serviços de cada componente. Estes testes utilizam um *stub* do SWiPE BE, com o objetivo de testar o processamento da informação que é feito nesta aplicação (Figura 57).

```
it('getAllSchoolYears() should call request get', () => {
  spyOn(requestService, 'get');
  service.getAllSchoolYears();
  expect(requestService.get).toHaveBeenCalled();
});
```

Figura 57 – Exemplo de teste realizado a uma função de um serviço

Os testes unitários do SWiPE FE foram desenvolvidos com recurso à *framework* Javascript Jasmine (Labs, 2010).

6.4.2 **Testes de integração**

Os testes de integração têm como objetivo testar as interações entre os diferentes componentes, focando-se sobretudo nas interfaces entre eles (ISTQB, 2017b). Este tipo de testes permite incluir: bases de dados, API, interfaces, etc. Apesar de tudo, estes testes podem também fazer uso de *stubs* e *drivers*. Existem dois níveis de testes de integração (ISTQB, 2018):

- Testes de integração de componentes, realizados ao nível do sistema interno, isto é, apenas são tidas em conta as interfaces e componentes do sistema privado;
- Testes de integração de sistema, centram-se nas relações existentes com sistemas externos.

6.4.2.1 Testes de integração no SWiPE BE

No componente SWiPE BE foram realizados testes de integração de componentes (SWiPE BE + base de dados) e testes de integração de sistema (SWiPE BE + base de dados + SWiPE MS).

Assim, estes testes permitem verificar se as interfaces implementadas entre estes componentes funcionam da forma esperada.

A Figura 58 representa um teste de integração realizado no SWiPE BE.

```
1   @Test
2   public void createSchoolYearTest() {
3       SchoolYearRequest request = new SchoolYearRequest();
4       request.setName("2019/2020");
5       request.setOwner("admin1@email.com");
6
7       String result = schoolYearService.createSchoolYear(request, "admin1@email.com");
8
9       assertThat(HttpStatus.valueOf(result)).isEqualTo(HttpStatus.CREATED);
10      SchoolYearDTO responseSchoolYear = (SchoolYearDTO) request.getResponses().toArray()[0];
11      assertThat(responseSchoolYear.getName()).isEqualTo("2019/2020");
12  }
```

Figura 58 – Exemplo de teste de integração no SWiPE BE

Para a execução deste teste foi criado um *stub* da base de dados com alguma informação relevante (nomeadamente utilizadores), de forma a testar a interface de ligação entre esta e o SWiPE BE. Assim, o novo ano letivo é criado na base de dados teste e a informação relativa a esta ação é enviada para a SWiPE MS (Figura 59).

```
 [x] Sent'On Sun Oct 13 15:25:28 WEST 2019 user admin1@email.com do getSchoolYear with SchoolYe
arDTO(schoolYearId=null, name=2017/2018, owner=admin1@email.com, editions=[]).'
```

Figura 59 – Envio da ação para a SWiPE MS decorrente do exemplo da Figura 41

À semelhança dos testes unitários no SWiPE BE, os testes de integração foram implementados com recurso à *framework open source* JUnit (Gamma and Beck, 2019).

6.4.2.2 Testes de integração no SWiPE FE

Ao nível do componente SWiPE FE foram realizados testes de integração de componentes, testando assim a interface entre este componente e o SWiPE BE (Figura 60).

```
it('should have "schoolYears" populated ', () => {
  console.log(component.schoolYears);
  expect(component.schoolYears.length).toBeGreaterThan(0);
});
```

Figura 60 – Exemplo de teste de integração no SWiPE FE

Não foram realizados testes de integração de sistema, dada a complexidade da interface externa do componente de autenticação.

À semelhança dos testes unitários no SWiPE FE, os testes de integração foram implementados com recursos à *framework* Javascript Jasmine (Labs, 2010).

6.4.3 **Testes de sistema**

Neste tipo de testes, o sistema é testado como um todo. Podem ser baseados em definições de casos de uso ou outras especificações de alto nível. Nos testes de sistema devem ser incluídos os requisitos funcionais e não funcionais (ISTQB, 2017c).

Foram realizados dois tipos de testes:

- Testes/experiências de desempenho de sistema, descritas no capítulo 7;
- Testes *end-to-end*, em que se executam as funcionalidades de um extremo ao outro em condições normais de funcionamento, e sem uso de *stubs*, *mock* ou outros mecanismos de simulação de partes do sistema.

Quanto aos testes *end-to-end*, dada a dimensão do sistema desenvolvimento, optou-se por testar um sub-conjunto de funcionalidades do sistema, desta feita de manipulação da entidade "ano letivo":

- Consulta da lista de anos letivos disponíveis no sistema;
- Criação de um ano letivo;
- Edição do ano letivo anteriormente criado;
- Nova consulta da lista, de forma a verificar a existência do novo ano letivo.

A Figura 61 representa o caso de teste de criação de um ano letivo.

```
it('create school years page', function() {
  browser.driver.sleep(2000);

  var menuButton = browser.driver.findElement(by.id('menuButton'));
  menuButton.click();

  browser.driver.sleep(1000);

  var schoolYearButton = browser.driver.findElement(by.id('schoolYears'));
  schoolYearButton.click();

  browser.driver.sleep(3000);

  var addNewSchoolYear = browser.driver.findElement(by.id('addNewSchoolYear'));
  addNewSchoolYear.click();

  browser.driver.sleep(3000);

  var schoolYearName = browser.driver.findElement(by.css("input[formControlName=name]"));
  schoolYearName.sendKeys('2019/2020');

  var saveSchoolYear = browser.driver.findElement(by.id('button-green'));
  saveSchoolYear.click();

});
```

Figura 61 – Teste de sistema – Criação de um ano letivo

Estes testes foram realizados com recurso à *framework* de Angular Protractor (Protractor, 2019).

6.4.4 **Testes de aceitação**

Os testes de aceitação têm como objetivo testar o sistema como um todo, incluindo os requisitos funcionais e não funcionais. Apesar de existirem diversos tipos de testes de aceitação, e.g. (i) testes de aceitação do utilizador, (ii) testes de aceitação operacional, (iii) teste alfa e beta, etc.). Neste contexto, foram realizados testes de aceitação de utilizador.

Assim, foram realizados os seguintes casos de teste, onde se assume que já existiam algumas configurações prévias do sistema, com vista à simplificação da execução deste conjunto de testes (Tabela 12):

Tabela 12 – Testes de aceitação

Critério	**Resultado obtido**
Autenticação no sistema com diversos papéis de utilizadores (Administrador, Docente, Externo e Estudante)	Cumpre
Criação de uma configuração de proposta, com campos em diversos idiomas com o perfil de administrador	Cumpre
Verificação da adequação da interface gráfica ao idioma favorito de cada um destes utilizadores.	Cumpre
Criação de proposta, utilizando a configuração definida anteriormente com o perfil de externo	Cumpre
Realização das ações definidas na configuração com base nas permissões dos diferentes utilizadores.	Cumpre

7 Experiências e Avaliação

Este capítulo tem como objetivos descrever as experiências de avaliação realizadas à solução desenvolvida. Para isso, serão descritas:

- A abordagem utilizada para a realização da avaliação do software desenvolvido;
- As experiências realizadas e resultados obtidos;
- A análise aos resultados obtidos.

Ao nível dos requisitos funcionais, é possível utilizar testes determinísticos para comprovar a correta realização destes requisitos (cf. secção 6.4). Por outro lado, no caso dos requisitos não funcionais, existem alguns em que é necessária a realização de experiências, que não testes determinísticos, como é o caso de (i) usabilidade e (ii) desempenho.

Uma vez que, a aplicação SWiPE FE foi desenvolvida como prova de conceito da aplicação SWiPE BE, não foram realizadas experiências de avaliação da usabilidade desta aplicação. Contudo, foram realizadas experiências de avaliação do desempenho do SWiPE BE, como estipulado na secção 3.4.5.

7.1 Objetivos

Como descrito na Secção 3.4.5, o desempenho do SWiPE BE considerado aceitável quando o tempo de resposta do sistema até 1 segundo (Card et al., 1991) em condições realistas de utiização do sistema pelos utilizadores potenciais.

Assim, o objetivo é determinar quais os limites de utilização até aos quais o servidor responde aos pedidos até 1 segundo.

7.2 Abordagem de avaliação

A abordagem seguida na avaliação do SWiPE BE foi a simulação de execução de pedidos à API REST do SWiPE BE em quantidades progressivamente maiores, de forma a tentar determinar os limites da utilização dentro dos parâmetros desejados.

Dada a população que frequenta as UC de projeto do ISEP (cf. secção 1.1) é de esperar que o sistema seja capaz de responder a picos de utilização destes utilizadores. Contudo, estes picos de utilização acontecem previsivelmente em momentos diferentes, visto que cada uma das UC tem as suas datas limites diferentes, como por exemplo de (i) submissão e (ii) escolha de propostas, (iii) de entrega de relatórios e (iv) avaliações. Contudo, curricularmente, estas UC ocorrem significativamente mais no segundo semestre do que no primeiro semestre ou anualmente.

Consideraram-se as seguintes assunções:

- Número máximo de UC em funcionamento simultâneo: 12;
- Número médio de estudante em simultâneo em todas as UC de projeto-estágio: 1500;
- Número máximo de estudantes a aceder em simultâneo ao sistema: 200[18].

De forma a determinar os pedidos REST para enviar para o SWiPE BE, foram executados os seguintes pedidos manuais reais de obtenção da configuração de uma proposta e de decisão sobre uma proposta. Estes pedidos têm a seguinte estrutura:

- Obtenção da configuração de uma proposta, constituído por três pedidos (Anexo G - Resultados dos pedidos REST):
    - /schoolyear/1/edition/1/proposalconfig/1/phaseconfig
    - /schoolyear/1/edition/1/proposalconfig/1/phaseconfig/2/ sectionconfig/
    - /schoolyear/1/edition/1/proposalconfig/1/phaseconfig/2/ sectionconfig/2/fieldconfig/
- Decisão sobre a proposta, constituído por três pedidos (Anexo G – Resultados dos pedidos REST):
    - /schoolyear/1/edition/1/proposal/2/phase/3
    - /schoolyear/1/edition/1/proposal/2/phase/3/section/8
    - /schoolyear/1/edition/1/proposal/2/phase/3/section/8/field/16

[18] 200 estudantes em utilização simultânea parece ser um número superior a uma utilização realista.

Para envio repetido dos pedidos para o servidor, a escolha do cliente HTTP recaiu no software JMeter (Foundation, 1998), que permite a realização de pedidos repetidos e o controlo e verificação de receção.

O SWiPE BE foi implantado numa máquina virtual do *data center* do DEI-ISEP, com as seguintes caraterísticas:

- Número de V-Cores: 6
- Quantidade de memória: 8 GB
- Tipo de disco: SCSI

A execução dos testes foi realizada a partir da máquina de desenvolvimento, o que pode prejudicar a comunicação com a máquina virtual.

7.3 Descrição das experiências

Foram realizadas dois grupos de experiências que simulam um número variável de utilizadores a executarem pedidos reais ao SWiPE BE:

- Leitura da configuração de uma proposta de projeto;
- Decisão sobre proposta de projeto.

7.3.1 Leitura de configuração de uma proposta

A Tabela 13 representa as experiências que foram realizadas na leitura (GET) de configuração de uma proposta:

Tabela 13 – Sistematização das experiências - Obtenção da configuração de uma proposta

Nº experiência	**Nº de utilizadores**	**Nº total de pedidos**
1	30 utilizadores (3 utilizadores/segundo)	90
2	50 utilizadores (5 utilizadores/segundo)	150
3	100 utilizadores (10 utilizadores/segundo)	300
4	200 utilizadores (20 utilizadores/segundo)	600
5	50 utilizadores (5 utilizadores/segundo), repetem os pedidos 3 vezes	450
6	100 utilizadores (10 utilizadores/segundo), repetem os pedidos 3 vezes	900
7	200 utilizadores (20 utilizadores/segundo), repetem os pedidos 3 vezes	1800

7.3.1.1 Resultados obtidos

A experiência 1 simulou a utilização do sistema por 30 utilizadores, distribuídos em intervalos de 10 segundos, produzindo assim, uma taxa de 3 utilizadores por segundo. Observando a Figura 62 (unidade dos valores em milissegundos) é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

Figura 62 – Boxplot da experiência 1 - Obtenção da configuração de uma proposta

A experiência 2 simulou a utilização de 50 utilizadores, distribuídos em intervalos de 10 segundos, produzindo assim, uma taxa de 5 utilizadores por segundo. Observando a Figura 63 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

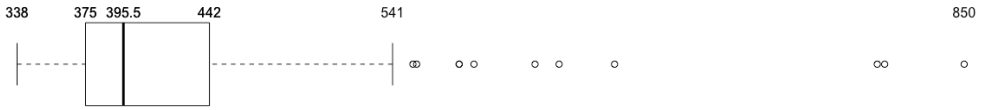

Figura 63 – Boxplot da experiência 2 - Obtenção da configuração de uma proposta

A experiência 3 simulou a utilização de 100 utilizadores, distribuídos em intervalos de 10 segundos, resultando assim em numa taxa de 10 utilizadores por segundo. Observando a Figura 64 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

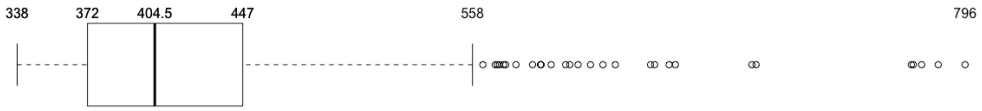

Figura 64 – Boxplot da experiência 3 - Obtenção da configuração de uma proposta

A experiência 4 simulou a utilização de 200 utilizadores, distribuídos em intervalos de 10 segundos, resultando assim em numa taxa de 20 utilizadores por segundo. Observando a Figura 65 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

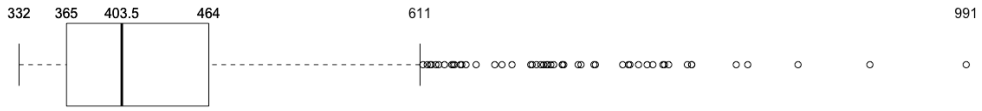

Figura 65 – Boxplot da experiência 4 - Obtenção da configuração de uma proposta

A experiência 5 simulou a utilização de 50 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 66 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

Figura 66 – Boxplot da experiência 5 - Obtenção da configuração de uma proposta

A experiência 6 simulou a utilização de 100 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 67 é possível concluir que nem todos os pedidos foram respondidos num segundo, mas que a tal falha ocorreu duas vezes, pelo que o problema poderá não ser da responsabilidade do software, mas de uma situação momentânea e excecional. Contudo, e porque ocorreu numa das experiências de maior carga, é também possível que se esteja perto do limite de desempenho desejado do software+hardware.

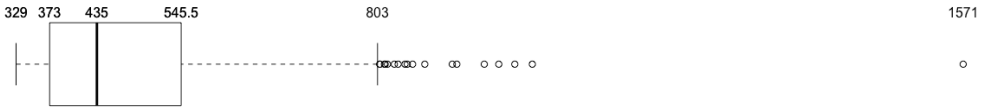

Figura 67 – Boxplot da experiência 6 - Obtenção da configuração de uma proposta

A experiência 7 simulou a utilização de 200 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 68 é possível concluir que nem todos os pedidos foram respondidos num segundo, existindo mesmo pedidos a demorar 4s. No total cerca de 15% dos pedidos não foram respondidos em menos de 1s pelo que se considera que possivelmente foi atingido o limite de desempenho desejado no software+hardware.

Figura 68 – Boxplot da experiência 7 - Obtenção da configuração de uma proposta

7.3.1.2 Interpretação dos resultados obtidos

As experiências realizadas permitem concluir que o sistema é capaz de responder corretamente até um determinado número de utilizadores (100 ou 200) a realizar o conjunto de pedidos definidos várias vezes (3), o que na realidade se irá verificar poucas ou nenhumas vezes. Assim, do ponto de vista de utilização do SWiPE BE, é possível garantir um funcionamento durante os períodos de maior afluência de submissão de propostas.

7.3.2 **Decisão sobre a proposta**

A Tabela 14 representa as experiências que foram realizadas sobre a funcionalidade de "decisão sobre a proposta" (PUT):

Tabela 14 – Sistematização das experiências – Decisão sobre a proposta

Nº experiência	**Nº de utilizadores**	**Nº total de pedidos**
1	30 utilizadores (3 utilizadores/segundo)	90
2	50 utilizadores (5 utilizadores/segundo)	150
3	100 utilizadores (10 utilizadores/segundo)	300
4	200 utilizadores (20 utilizadores/segundo)	600
5	50 utilizadores (5 utilizadores/segundo), repetem os pedidos 3 vezes	450
6	100 utilizadores (10 utilizadores/segundo), repetem os pedidos 3 vezes	900
7	200 utilizadores (20 utilizadores/segundo), repetem os pedidos 3 vezes	1800

7.3.2.1 Resultados obtidos

A experiência 1 simulou a utilização do sistema por 30 utilizadores, distribuídos em intervalos de 10 segundos, produzindo assim, uma taxa de 3 utilizadores por segundo. Observando a Figura 69 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

Figura 69 – Boxplot da experiência 1 - Decisão sobre a proposta

A experiência 2 simulou a utilização de 50 utilizadores, distribuídos em intervalos de 10 segundos, produzindo assim, uma taxa de 5 utilizadores por segundo. Observando a Figura 70 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

Figura 70 – Boxplot da experiência 2 - Decisão sobre a proposta

A experiência 3 simulou a utilização de 100 utilizadores, distribuídos em intervalos de 10 segundos, resultando assim em numa taxa de 10 utilizadores por segundo. Observando a Figura 71 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

Figura 71 – Boxplot da experiência 3 - Decisão sobre a proposta

A experiência 4 simulou a utilização de 200 utilizadores, distribuídos em intervalos de 10 segundos, resultando assim em numa taxa de 20 utilizadores por segundo. Observando, a Figura 72 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

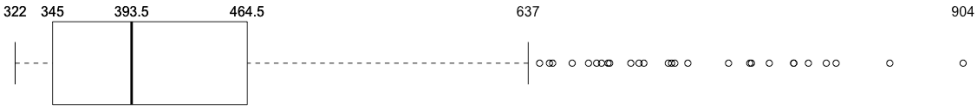

Figura 72 – Boxplot da experiência 4 - Decisão sobre a proposta

A experiência 5 simulou a utilização de 50 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 73 é possível concluir que todos os pedidos se realizaram em menos de 1 segundo.

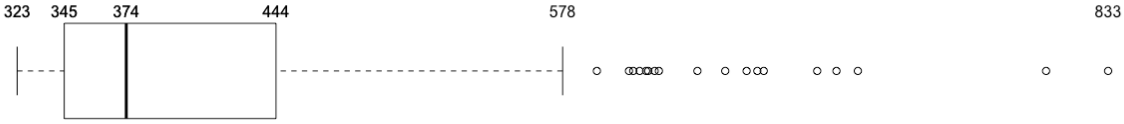

Figura 73 – Boxplot da experiência 5 - Decisão sobre a proposta

A experiência 6 simulou a utilização de 100 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 74 é possível concluir que nem todos os pedidos foram respondidos num segundo. Mais uma vez, a falha ocorreu numa das experiências de maior carga, pelo que é possível que se esteja perto do limite de desempenho desejado do software+hardware.

Figura 74 – Resultados da experiência 6 - Decisão sobre a proposta

A experiência 7 simulou a utilização de 200 utilizadores, distribuídos em intervalos de 10 segundos e fazendo com que estes realizem os pedidos 3 vezes. Observando a Figura 75, é possível concluir que nem todos os pedidos foram respondidos num segundo, existindo inclusive um que demorou cerca de 5s. Mais uma vez, a falha ocorreu na experiência de maior carga, pelo que é possível que se tenha atingido o limite de desempenho desejado do software+hardware.

Figura 75 – Resultados da experiência 7 – Decisão sobre a proposta

7.3.2.2 Interpretação dos resultados

As experiências realizadas permitem concluir que o sistema é capaz de responder corretamente até um determinado número de utilizadores (100 ou 200) a realizar o conjunto de pedidos definidos várias vezes (3), o que na realidade se irá verificar poucas ou nenhumas vezes. Assim, do ponto de vista de utilização do SWiPE BE, é possível garantir um funcionamento durante os períodos de maior afluência de submissão e decisão sobre propostas.

7.4 Síntese

Os resultados obtidos nas experiências permitem garantir um nível bastante elevado de desempenho em situações de carga elevada.

8 Conclusões

Neste capítulo são apresentadas as conclusões sobre o projeto desenvolvido, nomeadamente os objetivos alcançados, mas também as limitações encontradas e as melhorias que podem ser realizadas no futuro. Por fim, é realizada uma apreciação global do projeto.

8.1 Objetivos alcançados

O objetivo principal deste projeto visava o desenvolvimento de um sistema de software, capaz de responder às necessidades de gestão de projetos dos cursos do ISEP, garantindo o cumprimento de leis e regulamentos aplicáveis e suportando os diferentes processos adotados nestas UC.

O sistema implementado permite que cada UC configure o seu processo de gestão, podendo ser diferente em cada uma delas. O desenvolvimento de um protótipo, como prova de conceito do funcionamento do servidor desenvolvido permite, assim, comprovar que este objetivo foi cumprido.

A avaliação realizada ao servidor e a utilização de algumas ferramentas permitiram cumprir os objetivos de confiabilidade e manutenibilidade do sistema e código produzido. Desta forma, é possível que futuros desenvolvedores consigam compreender a arquitetura do sistema, mas também a implementação das funcionalidades.

Pode, por isso, considerar-se que os objetivos deste projeto foram cumpridos, mas existem requisitos (funcionais e não funcionais) que deverão ser alvo de trabalho futuro.

8.2 Limitações

Ao nível das limitações é de salientar, que apesar destas não terem tido uma influência negativa no cumprimento dos objetivos, influenciaram o desenvolvimento do projeto.

O facto da equipa de desenvolvimento do software ser apenas constituída por um elemento, não permitiu uma validação por parte de outras pessoas, existindo assim, alguns problemas no cumprimento de padrões e boas práticas, que tiveram de ser tidos em atenção numa fase mais avançada do projeto.

Também o facto do projeto não possuir requisitos bem definidos desde o início, traduziu-se numa alteração dos existentes ao longo do desenvolvimento. Esta alteração, muitas vezes complicou o trabalho desenvolvido até então.

8.3 Trabalho futuro

Ao nível do trabalho futuro, espera-se o desenvolvimento de uma aplicação de *frontend* que tenha em atenção os princípios de usabilidade descritos neste documento e outros, que possam ser adicionados através do contacto com as partes interessadas deste software. Também é esperado que existam apresentações regulares do software produzido aos RUC para que estes possam redefinir requisitos e emitir opiniões.

No futuro, deve ser estudada uma forma de interligar este sistema com o Portal do ISEP, o que não foi possível durante o desenvolvimento deste.

8.4 Apreciação final

O desenvolvimento deste projeto foi uma mais valia tanto a nível pessoal, como profissional, mas penso que também para o ISEP será um ponto de partida para uma melhor gestão das UC de projeto dos cursos deste instituto.

Para o ISEP e para as partes interessadas na gestão das UC de projeto, o software desenvolvido é um ponto de partida para a melhoria do funcionamento destas UC, podendo refletir-se numa melhoria das classificações dos estudantes, mas também das apreciações das organizações externas.

Ao nível profissional, este projeto permitiu-me aplicar diversos conhecimentos adquiridos ao longo da Licenciatura e Mestrado em Engenharia Informática, permitindo melhorar as capacidades de desenvolvimento de software, mas também a capacidade de sistematização, análise e desenho.

As decisões tomadas ao longo do desenvolvimento do projeto e a necessidade de estas serem criticamente discutidas, permitiu desenvolver esta capacidade pessoal. Para além disso, a necessidade de desenvolver um sistema destas dimensões sozinha, obrigou sempre a uma gestão responsável do tempo.

Por fim, considero que o resultado deste projeto foi satisfatório, visto que as limitações encontradas foram ultrapassadas com sucesso, não colocando em causa o cumprimento dos objetivos propostos. Desejo que, as experiências vividas ao longo do desenvolvimento deste projeto possam ser aplicadas no meu futuro profissional, nomeadamente todos os novos conceitos que aprendi.

Referências

Anastasov, M., 2019. CI/CD Pipeline: A Gentle Introduction [WWW Document]. Semaphore. URL https://semaphoreci.com/blog/cicd-pipeline (accessed 9.29.19).

Apache, 2019a. Apache Kafka [WWW Document]. Apache Kafka. URL https://kafka.apache.org/ (accessed 9.28.19).

Apache, 2019b. Apache RocketMQ [WWW Document]. Apache Rocket. URL https://rocketmq.apache.org/ (accessed 9.28.19).

Atlassian, 2019. Get started with Bitbucket Pipelines - Atlassian Documentation [WWW Document]. URL https://confluence.atlassian.com/bitbucket/get-started-with-bitbucket-pipelines-792298921.html (accessed 9.29.19).

Bachmann, F., Bass, L., Clements, P.C., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R., Stafford, J.A., 2010. Documenting Software Architectures: Views and Beyond.

Beck, K., 2000. Test-Driven Development By Example.

Berners-Lee, T., 1994. World Wide Web Consortium (W3C) [WWW Document]. URL https://www.w3.org/ (accessed 10.8.19).

Brown, S., 2011. The C4 model for visualising software architecture [WWW Document]. URL https://c4model.com/ (accessed 10.4.19).

Burn, O., 2001. checkstyle – Checkstyle 8.25 [WWW Document]. URL https://checkstyle.sourceforge.io/ (accessed 10.13.19).

Chesky, B., Gebbia, J., Blecharczyk, N., 2008. Alugueres de Férias, Casas, Experiências e Lugares [WWW Document]. Airbnb. URL https://www.airbnb.pt/ (accessed 10.13.19).

Comissão Europeia, 2016. Para que serve o Regulamento Geral sobre a Proteção de Dados (RGPD)? [WWW Document]. Com. Eur. - Eur. Comm. URL https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-does-general-data-protection-regulation-gdpr-govern_pt (accessed 8.17.19).

Corporation, R.S., 2001a. Activity: Prioritize Use Cases [WWW Document]. URL https://sceweb.uhcl.edu/helm/RationalUnifiedProcess/process/activity/ac_priuc.htm#Prioritize%20Use%20Cases%20and%20Scenarios (accessed 10.5.19).

Corporation, R.S., 2001b. Guidelines: Requirements Management Plan [WWW Document]. URL https://sceweb.uhcl.edu/helm/RationalUnifiedProcess/process/modguide/md_rmp.htm#benefit (accessed 10.5.19).

Crockford, D., 2002. TSLint [WWW Document]. URL https://palantir.github.io/tslint/ (accessed 10.13.19).

DEI, 2019a. Projetos/Estágios - DEI [WWW Document]. URL http://www4.dei.isep.ipp.pt/gestproj/ (accessed 5.1.19).

DEI, 2019b. TESES/PROJETOS - DEI [WWW Document]. URL https://www4.dei.isep.ipp.pt/gestmei/index.php (accessed 5.1.19).

Diário da República Eletrónico, 2018a. Decreto-Lei 65/2018 [WWW Document]. Diário Repúb. Eletrónico. URL https://dre.pt/web/guest/pesquisa/-/search/116068879/details/maximized (accessed 12.28.18).

Diário da República Eletrónico, 2018b. Resolução do Conselho de Ministros 41/2018, 2018-03-28 [WWW Document]. URL https://dre.pt/home/-/dre/114937034/details/maximized (accessed 2.21.19).

Eeles, P., 2005. Capturing Architectural Requirements [WWW Document]. URL https://www.ibm.com/developerworks/rational/library/4706.html#N100A7 (accessed 1.30.19).

Facebook Inc., 2013. React – A JavaScript library for building user interfaces [WWW Document]. URL https://reactjs.org/ (accessed 10.7.19).

Fielding, R.T., 2000. Architectural Styles and the Design of Network-based Software Architectures 180.

Firesmith, D., 2013. Using V Models for Testing [WWW Document]. URL https://insights.sei.cmu.edu/sei_blog/2013/11/using-v-models-for-testing.html (accessed 10.5.19).

Foundation, A.S., 1998. Apache JMeter - Apache JMeter™ [WWW Document]. URL https://jmeter.apache.org/ (accessed 10.10.19).

Foundation, E., 2019. The AspectJ Project | The Eclipse Foundation [WWW Document]. URL https://www.eclipse.org/aspectj/ (accessed 8.28.19).

Foundation, N. js, 2019. Node.js [WWW Document]. Node.js. URL https://nodejs.org/en/ (accessed 9.27.19).

Fowler, M., 2010. Richardson Maturity Model [WWW Document]. martinfowler.com. URL https://martinfowler.com/articles/richardsonMaturityModel.html (accessed 10.12.19).

Fowler, M., 2006. GUI Architectures [WWW Document]. martinfowler.com. URL https://martinfowler.com/eaaDev/uiArchs.html (accessed 10.9.19).

Fowler, M., 2005a. Domain Event [WWW Document]. martinfowler.com. URL https://martinfowler.com/eaaDev/DomainEvent.html (accessed 10.4.19).

Fowler, M., 2005b. Event Sourcing [WWW Document]. martinfowler.com. URL https://martinfowler.com/eaaDev/EventSourcing.html (accessed 9.23.19).

Fowler, M., 2004. Audit Log [WWW Document]. martinfowler.com. URL https://martinfowler.com/eaaDev/AuditLog.html (accessed 10.4.19).

Fowler, M., 2002. Patterns of Enterprise Application Architecture.

Gamma, E., Beck, K., 2019. JUnit 5 [WWW Document]. URL https://junit.org/junit5/ (accessed 10.5.19).

Gestão de projetos curriculares [WWW Document], 2019. . Google Docs. URL https://docs.google.com/forms/d/e/1FAIpQLSeqvknKugNYF_uT9f5VUk33rhtsTRuiyG2UpCjiIwreWyxQHQ/viewform?usp=embed_facebook (accessed 2.2.19).

Google, 2010. Angular [WWW Document]. URL https://angular.io/ (accessed 10.7.19).

Hakansson, M., 2019. Why Attribute-Based Access Control Will Become the Standard Model for Large Enterprises - DZone Security [WWW Document]. dzone.com. URL https://dzone.com/articles/access-control-101-why-attribute-based-access-cont (accessed 10.4.19).

Hauser, J.R., Clausing, D., 1988. The House of Quality. Harv. Bus. Rev.

Hohpe, G., Woolf, B., 2003. Enterprise Integration Patterns - Message Store [WWW Document]. URL https://www.enterpriseintegrationpatterns.com/patterns/messaging/MessageStore.html (accessed 9.23.19).

IAPMEI, 2018. IAPMEI - Regulamento Geral de Proteção de Dados [WWW Document]. URL https://www.iapmei.pt/PRODUTOS-E-SERVICOS/Assistencia-Tecnica-e-Formacao/Regime-Geral-de-Protecao-de-Dados.aspx (accessed 11.29.18).

IBM, 2019. Auditing, logging, and monitoring [WWW Document]. URL https://cloud.ibm.com/docs/services/Db2onCloud?topic=Db2onCloud-aud-log-mon (accessed 10.13.19).

IBM, 2014a. Authentication and authorization [WWW Document]. URL https://www.ibm.com/support/knowledgecenter/en/SSEUEX_3.0.5/com.ibm.installingeuc.doc/eucpl050.htm (accessed 2.20.19).

IBM, 2014b. Traceability [WWW Document]. URL https://www.ibm.com/support/knowledgecenter/en/SSYMRC_6.0.4/com.ibm.rational.rrm.help.doc/topics/c_trace.html (accessed 2.20.19).

IBM, 2014c. Service-oriented architecture (SOA) [WWW Document]. URL www.ibm.com/support/knowledgecenter/en/ssmq79_9.5.1/com.ibm.egl.pg.doc/topics/pegl_serv_overview.html (accessed 9.27.19).

ISEP, 2019a. Portal ISEP [WWW Document]. URL https://portal.isep.ipp.pt/intranet/conta/AreaDeTrabalho.aspx (accessed 2.2.19).

ISEP, 2019b. Moodle ISEP [WWW Document]. MoodleISEP. URL https://moodle.isep.ipp.pt/ (accessed 5.1.19).

ISEP, 2018a. Tornar a instituição ainda mais forte [WWW Document]. URL http://www.isep.ipp.pt/New/ViewNew/5675 (accessed 2.9.19).

ISEP, 2018b. Nova geração do PRAXIS facilita oferta e procura de estágios [WWW Document]. URL http://www.isep.ipp.pt/New/ViewNew/5748 (accessed 1.28.19).

ISTQB, 2018. What are Software Testing Levels? [WWW Document]. URL http://tryqa.com/what-are-software-testing-levels/ (accessed 10.5.19).

ISTQB, 2017a. What is Component testing? [WWW Document]. URL http://tryqa.com/what-is-component-testing/ (accessed 10.5.19).

ISTQB, 2017b. What is Integration testing? Examples, How To Do, Types/Approaches, Differences [WWW Document]. URL http://tryqa.com/what-is-integration-testing/ (accessed 10.5.19).

ISTQB, 2017c. What is System testing? [WWW Document]. URL http://tryqa.com/what-is-system-testing/ (accessed 10.5.19).

Jameson, A., 2001. Designing User-Adaptive Systems 3.

Koen et al., P.A., 2004. Fuzzy Front End: Effective Methods, Tools, and Techniques 5–35.

Kruchten, P.B., 1995. The 4+1 View Model of architecture.

Labs, P., 2010. Jasmine Documentation [WWW Document]. URL https://jasmine.github.io/index.html (accessed 10.5.19).

Larman, C., 2005. Applying UML and patterns: an introduction to object-oriented analysis and design and iterative development, 3rd ed. ed. Prentice Hall PTR, c2005, Upper Saddle River, N.J.

Lesh, B., Driscoll, D., Wortmann, J.-N., Jamieson, N., Kwon, O., Taylor, P., Lee, T., 2019. RxJS [WWW Document]. URL https://rxjs-dev.firebaseapp.com/ (accessed 9.29.19).

Linnenfelser, M., Weber, S., Rech, J., 2010. An Overview of and Criteria for the Differentiation and Evaluation of RIA Architectures. IGI Global. https://doi.org/10.4018/978-1-60566-384-5

Mandacina, B., 2016. Configurable vs. Customizable. URL http://locktonbenefitsblog.com/configurable-vs-customizable/ (accessed 10.7.19).

Martin, R.C., 2000. Design Principles and Design Patterns 34.

Microsoft, 2019a. Azure Active Directory | Microsoft Azure [WWW Document]. URL https://azure.microsoft.com/pt-pt/services/active-directory/ (accessed 9.28.19).

Microsoft, 2019b. O que é o RBAC (controle de acesso baseado em função) para recursos do Azure? [WWW Document]. URL https://docs.microsoft.com/pt-pt/azure/role-based-access-control/overview (accessed 10.4.19).

Microsoft, 2019c. Bases de dados NoSQL | Microsoft Azure [WWW Document]. URL https://azure.microsoft.com/pt-pt/overview/nosql-database/ (accessed 9.28.19).

Microsoft, 2018. Authentication - Windows applications [WWW Document]. URL https://docs.microsoft.com/en-us/windows/win32/secauthn/authentication-portal (accessed 10.7.19).

Microsoft, 2017. Add your own attributes to custom policies in Azure Active Directory B2C | Microsoft Docs [WWW Document]. URL https://docs.microsoft.com/bs-latn-ba/azure/active-directory-b2c/active-directory-b2c-create-custom-attributes-profile-edit-custom (accessed 10.8.19).

Nielsen, J., 2012. Usability 101: Introduction to Usability [WWW Document]. Nielsen Norman Group. URL https://www.nngroup.com/articles/usability-101-introduction-to-usability/ (accessed 2.23.19).

OASIS, 2001. OASIS eXtensible Access Control Markup Language (XACML) TC | OASIS [WWW Document]. URL https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml (accessed 10.13.19).

OMG, 2019. Welcome To UML Web Site! [WWW Document]. URL https://www.uml.org/ (accessed 8.17.19).

Oracle, 2019. What is a relational database? [WWW Document]. URL https://www.oracle.com/database/what-is-a-relational-database/ (accessed 9.28.19).

Oracle, 2006. Javadoc Tool Home Page [WWW Document]. URL https://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html (accessed 10.13.19).

Palermo, J., 2008. The Onion Architecture : part 1 [WWW Document]. Program. Palermo. URL https://jeffreypalermo.com/2008/07/the-onion-architecture-part-1/ (accessed 9.28.19).

Pan, J., 1999. Software Reliability [WWW Document]. URL https://users.ece.cmu.edu/~koopman/des_s99/sw_reliability/ (accessed 2.11.19).

Parnas, D.L., 1972. On the Criteria To Be Used in Decomposing Systems into Modules 15, 6.

Pivotal, 2019. Messaging that just works — RabbitMQ [WWW Document]. URL https://www.rabbitmq.com/ (accessed 8.28.19).

PRAXIS, 2019. Student internships and challenges in PRAXIS [WWW Document]. URL http://www.praxisnetwork.eu/ (accessed 5.1.19).

Preston-Werner, T., Wanstrath, C., Hyett, P.J., Chacon, S., 2008. Build software better, together [WWW Document]. GitHub. URL https://github.com (accessed 10.13.19).

Protractor, 2019. Protractor - end-to-end testing for AngularJS [WWW Document]. URL https://www.protractortest.org/#/ (accessed 10.5.19).

Rich, N., Holweg, M., 2000. INNOREGIO: dissemination of innovation and knowledge management techniques.

Rick-Anderson, 2019. ASP.NET Documentation [WWW Document]. URL https://docs.microsoft.com/en-us/aspnet/ (accessed 9.27.19).

Roberts, P., 2007. Quality Function Deployment.

Rouse, M., 2007. What is Rich Internet Application (RIA)? - Definition from WhatIs.com [WWW Document]. URL https://whatis.techtarget.com/definition/Rich-Internet-Application-RIA (accessed 9.27.19).

Saaty, T.L., 2008. Decision making with the analytic hierarchy process. Int. J. Serv. Sci. 1, 83.

Saternos, C., St. Laurent, S., MacDonald, A., Demarest, R., 2014. Client-server web apps with JavaScript and Java, First edition. ed. O'Reilly Media, Inc, Sebastopol, CA.

Software, P., 2019. Spring Projects [WWW Document]. URL https://spring.io/projects/spring-boot (accessed 8.28.19).

Subramanian, N., Chung, L., 2001a. Metrics for Software Adaptability 14.

Subramanian, N., Chung, L., 2001b. Software Architecture Adaptability: An NFR Approach 10.

Vanek, C., 2016. Customized vs. Configurable Software Solutions: Which Should You Choose? [WWW Document]. npENGAGE. URL https://npengage.com/nonprofit-technology/customized-vs-configurable-software-solutions-which-should-you-choose/ (accessed 2.9.19).

Visser, J., 2015. Building Maintainable Software.

Vocke, H., 2018. The Practical Test Pyramid [WWW Document]. martinfowler.com. URL https://martinfowler.com/articles/practical-test-pyramid.html (accessed 9.27.19).

Wasson, M., 2013. ASP.NET - Single-Page Applications: Build Modern, Responsive Web Apps with ASP.NET [WWW Document]. URL https://msdn.microsoft.com/en-us/magazine/dn463786.aspx (accessed 9.27.19).

Woodall, T., 2003. Conceptualising “Value for the Customer”: An Attributional, Structural and Dispositional Analysis. Acad. Mark. Sci. Rev. 12.

WSO2, 2019. XACML Architecture - Identity Server 5.3.0 - WSO2 Documentation [WWW Document]. URL https://docs.wso2.com/display/IS530/XACML+Architecture (accessed 9.28.19).

You, E., 2014. Vue.js [WWW Document]. URL https://vuejs.org/ (accessed 10.7.19).

Zeithaml, V., 1988. Consumer Perceptions of Price, Quality, and Value: A Means-End Model and Synthesis of Evidence. J. Mark. 52, 14.

Anexo A - Inquérito sobre Gestão de Projetos Curriculares

Gestão de projetos curriculares

Este questionário tem por objetivo conhecer e analisar os processos/tarefas de gestão de projetos curriculares dos cursos (licenciaturas e mestrados) do ISEP.
A resposta às questões deve referir-se apenas às informações do seu curso/UC. A análise dessas respostas condicionará o desenvolvimento de uma aplicação de software de apoio à gestão das tarefas desses projetos.
O preenchimento deste questionário tem uma duração de cerca de 10 minutos.

*Obrigatório

Geral

1. **Quantos professores se encontram envolvidos no processo de gestão da UC?** *
   Marcar apenas uma oval.
   - ◯ 1
   - ◯ 2-4
   - ◯ 5-8
   - ◯ +8

2. **O tempo de serviço docente atribuído à gestão da UC é suficiente?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Discordo totalmente	◯	◯	◯	◯	◯	Concordo totalmente

Propostas de Projetos

3. **Como podem os proponentes publicitar as propostas de projeto?** *
   Marcar tudo o que for aplicável.
   - ☐ Moodle
   - ☐ Portal
   - ☐ Praxis
   - ☐ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
   - ☐ Lista em papel
   - ☐ Email
   - ☐ Outra: _____

4. **As propostas são revistas/validadas pelo RUC?** *
   Marcar apenas uma oval.

   - ◯ Sim
   - ◯ Não

5. **Como podem os alunos consultar as propostas de projeto?** *
   Marcar tudo o que for aplicável.

   - ☐ Moodle
   - ☐ Portal
   - ☐ Praxis
   - ☐ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
   - ☐ Lista em papel
   - ☐ Email
   - ☐ Outra: _____

6. **Como é que os alunos contactam o proponente da proposta de projeto?** *
   Marcar tudo o que for aplicável.

   - ☐ Moodle
   - ☐ Portal
   - ☐ Praxis
   - ☐ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
   - ☐ Email
   - ☐ Telefone
   - ☐ Presencialmente
   - ☐ Outra: _____

Seleção e Atribuição

7. **Como é que os alunos selecionam/escolhem o projeto?** *
   Marcar tudo o que for aplicável.

   - ☐ Moodle
   - ☐ Portal
   - ☐ Praxis
   - ☐ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
   - ☐ Email
   - ☐ Telefone
   - ☐ Presencialmente
   - ☐ Outra: _____

8. **Como é comunicada a atribuição do projeto ao aluno?** *
   Marcar tudo o que for aplicável.

   - [ ] Moodle
   - [ ] Portal
   - [ ] Praxis
   - [ ] Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
   - [ ] Email
   - [ ] Telefone
   - [ ] Presencialmente
   - [ ] Outra: _____

9. **Em caso de vários alunos selecionarem o mesmo projeto, a quem é atribuído o projeto?** *
   Marcar apenas uma oval.

   - ◯ O proponente decide.
   - ◯ O RUC decide.
   - ◯ Aleatoriamente.
   - ◯ O 1º estudante a contactar o proponente é o escolhido.
   - ◯ Outra: _____

Formalização

10. **Existe um processo de formalização (acordo/contrato) de projeto entre estudante, proponente e ISEP?** *
    Marcar apenas uma oval.

    - ◯ Não *Passe para a pergunta 17.*
    - ◯ Sim, em papel
    - ◯ Sim, por email
    - ◯ Sim, presencialmente
    - ◯ Sim, por telefone
    - ◯ Sim, em formato digital
    - ◯ Outra: _____

Formalização

11. **Na formalização é definido/refinado o problema do projeto?** *
    Marcar apenas uma oval.

    - ◯ Sim
    - ◯ Não

12. **Na formalização são definidos/refinados os objetivos do projeto?** *
Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

13. **Na formalização é definida/refinada a abordagem a seguir pelo projeto?** *
Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

14. **Esta formalização é revista e devolvida aos intervenientes?** *
Marcar apenas uma oval.

- ◯ Sim é revista, mas não devolvida
- ◯ Sim é revista e devolvida
- ◯ Não é revista

15. **Qual o formato da formalização?** *
Marcar apenas uma oval.

- ◯ Preenchimento de documento de texto livre
- ◯ Preenchimento de documento de texto formatado
- ◯ Questionário
- ◯ Ata de reunião
- ◯ Outra: _____

16. **Onde é feita a submissão da formalização?** *
Marcar apenas uma oval.

- ◯ Moodle
- ◯ Portal
- ◯ Praxis
- ◯ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
- ◯ Email
- ◯ Outra: _____

Entregas periódicas/parciais

17. **Existem entregas periódicas/parciais** *
Marcar apenas uma oval.

- ◯ Sim
- ◯ Não *Passe para a pergunta 22.*

Entregas periódicas/parciais

18. **Quantas entregas periódicas/parciais existem?** *

_____

19. **Estas (todas ou algumas) entregas têm algum peso na nota final atribuída ao aluno?** *
Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

20. **Que tipo de submissão é efetuada?** *
Marcar tudo o que for aplicável.

- ☐ Preenchimento de documento de texto livre
- ☐ Preenchimento de documento de texto formatado
- ☐ Questionário
- ☐ Ata de reunião
- ☐ Outra: _____

21. **Onde são submetidas as entregas periódicas/parciais?** *
Marcar tudo o que for aplicável.

- ☐ Moodle
- ☐ Portal
- ☐ Praxis
- ☐ Plataforma de gestão de projetos (e.g. GestProj/GestMEI)
- ☐ Email
- ☐ Outra: _____

Entrega final

O relatório do projeto é (atualmente) submetido no Portal, dando início a um processo de aceitação por parte do orientador e marcação de prova por parte do RUC.

22. **O software de apoio a este processo é adequado?** *
Marcar apenas uma oval.

	1	2	3	4	5	
Discordo totalmente	◯	◯	◯	◯	◯	Concordo totalmente

Marcação de provas

23. **Como é escolhido o arguente?** *
Marcar tudo o que for aplicável.

- [ ] Por sugestão do orientador
- [ ] Por sugestão do estudante
- [ ] Por sugestão do proponente
- [ ] Definido exclusivamente pelo RUC ou por alguém por ele indicado
- [ ] Aleatoriamente
- [ ] Outra: _____

24. **Como é escolhido o presidente do júri?** *
Marcar tudo o que for aplicável.

- [ ] Por sugestão do orientador
- [ ] Por sugestão do estudante
- [ ] Por sugestão do proponente
- [ ] Definido pelo RUC ou por alguém por ele indicado
- [ ] Aleatoriamente
- [ ] Outra: _____

25. **Como é convidado/indicado/aceite o presidente e arguente do júri?** *
Marcar tudo o que for aplicável.

- [ ] Por email
- [ ] Por telefone
- [ ] Por portal
- [ ] Presencialmente
- [ ] Através da plataforma de gestão de projetos (GestProj/GestMEI)
- [ ] Outra: _____

26. **Quantos professores se encontram envolvidos no processo de avaliação de projetos?** *
Marcar apenas uma oval.

- ◯ 6-10
- ◯ 11-20
- ◯ 21-35
- ◯ 36-50
- ◯ +50

Avaliação

Para além do processo de avaliação formal segundo as leis e regulamentos em vigor, alguns cursos aproveitam a oportunidade da (tipicamente) última avaliação do estudante, para fazer inquéritos a estudante e avaliadores.

27. **É feito algum inquérito no final da prova?** *
Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

28. **Em caso afirmativo, a quem?**
Marcar tudo o que for aplicável.

- ☐ Estudante
- ☐ Arguente
- ☐ Proponente/Empresa
- ☐ Orientador
- ☐ Presidente do Júri
- ☐ Outra: _____

29. **Em caso afirmativo, de que tipo?**
Marcar tudo o que for aplicável.

- ☐ De softskills
- ☐ De hardskills
- ☐ De outcomes (objetivos/resultados expectáveis) do curso
- ☐ De apreciação (subjetiva) do curso/departamento/ISEP
- ☐ De sugestões de melhoria curso/departamento/ISEP
- ☐ Outra: _____

Final

30. **Considera que existe software de apoio às tarefas de gestão da UC?** *
Marcar apenas uma oval.

- ◯ Sim, existe, mas apenas auxilia algumas tarefas
- ◯ Sim, existem vários de auxílio a todas as tarefas de forma integrada
- ◯ Sim, existem vários de auxílio a todas as tarefas, mas de forma não integrada
- ◯ Não existe nenhum software

31. **Qual a relevância de um software que permita a gestão integrada destes processos/tarefas?** *
Marcar apenas uma oval.

	1	2	3	4	5	
Pouco relevante	◯	◯	◯	◯	◯	Muito relevante

32. **Se desejar/permitir ser contactado, indique a sua sigla**

33. **Gostaria de deixar alguma sugestão para o desenvolvimento da nova aplicação?**

Anexo B – Atas das reuniões realizadas com os RUC

Ata de Reunião – Licenciatura em Eng. Mecânica

Participantes:

Joana Santos – 1140418

Nuno Silva – NPS

Isabel Sarmento – ISP

Rui Rego – RFR

Aos 6 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados na UC de projeto curricular na Licenciatura em Engenharia Mecânica.

Assim, foram fornecidas as seguintes informações e sugestões:

- Existe apenas uma percentagem dos alunos que frequentam a UC que realizam estágio em empresas.
- Estas empresas são angariadas pelos alunos ou pelos docentes responsáveis por tal (RFR).
- Os alunos emitem as suas preferências e, posteriormente são seriados, com recurso a determinados critérios (ECTS concluídos, média). Esta seriação e as preferências são envolvidas num processo de otimização que decide que projeto será atribuído a cada aluno.
- Dado que existem alunos a realizar o projeto em sala, não existe a submissão final no Portal, sendo que todo o processo é realizado no *Moodle*. A RUC não considera a configuração do *Moodle* algo simples.
- A apresentação final conta com a presença do orientador e do supervisor da empresa, mas também do arguente e da presidência do ISEP ou alguém por esta delegado.
- Não é preenchida nenhuma ata na prova final.

- É realizado um questionário sobre as *softskills* do estudante.
- A RUC sugeriu que o protocolo que é realizado aquando da formalização do estágio, entre empresa, estudante e ISEP, deveria ser em formato digital e não em papel.

Ata de Reunião – Licenciatura em Eng. Química e Biorrecursos

Participantes:

Joana Santos – 1140418

Nuno Silva – NPS

Isabel Pereira – IMP

Margarida Ribeiro – MGR

Aos 7 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados nas UC de projeto curricular nas Licenciaturas em Engenharia Química e Biorrecursos.

Assim, foram fornecidas as seguintes informações e sugestões:

- As empresas são contactadas pelas docentes, sendo que são necessários diversos contactos até que as propostas de estágio sejam realizadas.
- A divulgação das propostas é feita via email.
- A escolha dos estudantes é feita pelas empresas, mas são as docentes que escolhem os alunos que serão entrevistados, através de um processo de seriação.
- Existe uma apresentação intermédia, com vista ao registo do progresso feito pelo aluno. Esta apresentação não conta para a nota final atribuída ao estudante.
- O relatório é submetido no Portal ISEP e, posteriormente é enviado a um arguente.
- Existe o preenchimento de um questionário formatado no dia da prova, por parte da empresa onde o estudante realizou o estágio.
- O convite aos docentes que irão arguir a prova final é feito através de convite informal, muitas vezes são convidados presencialmente.
- Foi referido pelas docentes que por vezes, se esquecem de informar todas as pessoas envolvidas na avaliação da prova. Pelo que, mencionaram que o desenvolvimento de uma aplicação que permitisse enviar os emails sempre que existe uma marcação de prova, antes desta ser efetivada no Portal, podia ser relevante.

- As docentes acharam o desenvolvimento de um *software* que auxilie o processo de gestão algo importante, mas não pretendem que o seu processo atual seja alterado.

Ata de Reunião – Licenciatura em Eng. de Sistemas

Participantes:

Joana Santos – 1140418

Nuno Silva – NPS

Maria Teresa Costa – MCO

Aos 8 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados nas UC de projeto curricular na Licenciatura em Engenharia de Sistemas

Assim, foram fornecidas as seguintes informações e sugestões:

- A docente referiu a indisponibilidade caso a sugestão passasse por unificar as UC de projeto curricular.
- Existe uma dificuldade enorme, nas empresas criarem propostas de estágio. Pelo que a docente refere que não pretende que outros alunos possam consultar as propostas de estágios dos estudantes de Engenharia de Sistemas.
- Os estudantes têm total liberdade na escolha de projetos e orientadores.
- Não existe nenhum ponto de controlo oficial, apenas o que cada orientador decidir.
- A marcação de provas também não é um processo cognitivamente simples. A docente refere que se for desenvolvida uma aplicação que permita auxiliar este processo está interessada.
- A docente deixou como sugestão que um RUC deve puder rejeitar as provas quando estas são submetidas no Portal.

Ata de Reunião – Mestrado em Eng. Química

Participantes:

Joana Santos – 1140418

Nuno Silva – NPS

Gilberto Pinto – GAP

Aos 8 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados na UC de tese do Mestrado em Engenharia Química.

Assim, foram fornecidas as seguintes informações e sugestões:

- O RUC indicou que o processo de gestão de proposta é algo complicado, pois é necessário contactar empresas para que estas forneçam estágios ou que os estudantes contactem diretamente as empresas em que pretendem estagiar.
- As propostas são validadas, no ponto de vista do problema que pretendem resolver e se este tem relevância para o curso.
- A gestão da atribuição das propostas aos alunos é realizada em Excel, sendo que é necessário gerir toda esta informação.
- Os alunos são alvo de um processo de seriação que cumpre diversos critérios e são estes critérios que definem quem realiza o estágio, caso a empresa não pretenda ter influência neste processo.
- Existem relatório de progresso que são avaliados e controlados pelo orientador do projeto.
- O processo de avaliação final segue diversos critérios que tem em atenção o trabalho realizado pelo estudante ao longo do decurso do desenvolvimento do projeto.
- O júri da prova é constituído pelos orientadores do projeto, pelo arguente e pelo presidente do júri.
- Os diferentes membros do júri avaliam diferentes critérios, conforme a função que desempenham.

Ata de Reunião – Licenciatura e Mestrado em Eng. Civil

Participantes:

Joana Santos – 1140418

Teresa Neto – TIS

Jorge Mendes – JJM

Aos 12 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados nas UC de projeto-estágio e tese do Departamento de Engenharia Civil.

Assim, foram fornecidas as seguintes informações e sugestões:

- Na licenciatura, os estudantes podem optar por realizar o estágio em ambiente empresarial ou realizar um projeto, em grupos de 2 com um orientador do ISEP.
- No mestrado, os projetos são realizados individualmente em ambiente empresarial ou em projeto.
- A proposta de estágio é feita através de um ficheiro formatado, sendo preenchido pelo orientador, proponentes ou pelo RUC em contacto direto com a empresa.
- Os docentes indicaram que a existência de uma plataforma de gestão de propostas poderia ser mais vantajosa no mestrado, uma vez que na licenciatura existe um contacto mais direto com as empresas e a maioria dos alunos realiza os projetos no ISEP.
- Os alunos são seriados através de diversos critérios, nomeadamente média e ETCS concluídos.
- Os docentes sugeriram a possibilidade da candidatura dos alunos, logo o processo de seriação ser efetuado pela plataforma e não manualmente como atualmente.
- Não existem entregas parciais ao longo do desenvolvimento do projeto, o orientador é que é responsável por verificar o estado do projeto.
- Os docentes indicaram que reunir o júri e verificar as disponibilidades dos envolvidos é, por vezes, muito complicado, pelo que o desenvolvimento de uma aplicação que auxilie este processo seria uma mais valia.

- No mestrado, os arguentes podem ser convidados externamente, pelo que é um processo demorado e que com *software* de apoio poderia ser facilitado.

Ata de Reunião – Licenciatura em Eng. Eletrotécnica – Sistemas Elétricos de Energia

Participantes:

Joana Santos – 1140418

Nuno Silva - NPS

Maria Judite Ferreira – MJU

Aos 12 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados na UC de projeto-estágio da Licenciatura em Engenharia Eletrotécnica – Sistemas Elétricos de Energia.

Assim, foram fornecidas as seguintes informações e sugestões:

- Os estudantes podem realizar projeto ou estágios em ambiente empresarial.
- A responsabilidade da escolha do projeto e do orientador é do estudante.
- Quando são submetidas as propostas, é entregue um documento onde estão explicitados o problema, abordagem, etc.
- Não existem entregas parciais, os orientadores têm a responsabilidade de controlar o processo de desenvolvimento do projeto.
- Os docentes são convocados para arguir as apresentações, pelo que não existem convites formais.

Ata de Reunião – Licenciatura em Engenharia e Gestão Industrial

Participantes:

Joana Santos – 1140418

Nuno Silva - NPS

Paulo Ávila – PSA

Aos 12 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados na UC de projeto-estágio da Licenciatura em Engenharia e Gestão Industrial.

Assim, foram fornecidas as seguintes informações e sugestões:

- As empresas preenchem um questionário (Google Forms) onde indicam as características da proposta de estágio
- Os estudantes podem realizar projetos ou estágios em ambiente empresarial.
- São os estudantes que escolhem as propostas e se candidatam, sendo que são os proponentes que escolhem. Após a escolha, preenchem um questionário (Google Forms) onde indicam as características do estágio que vão realizar
- É guardada em ficheiro Excel, a informação referente ao aluno e à proposta na qual foi colocado.
- Ao longo do ano é realizada uma avaliação qualitativa, que apesar de não possuir um peso específico na avaliação final, permite que a nota seja de alguma forma influenciada.
- Os orientadores podem escolher os arguentes das provas, sendo que não existe um processo muito pormenorizado nesta tarefa.

Ata de Reunião – Licenciatura em Eng. Eletrotécnica e de Computadores

Participantes:

Joana Santos – 1140418

Nuno Silva - NPS

Francisco Pereira – FDP

Aos 14 dias do mês de fevereiro do ano de 2019, foi realizada uma reunião entre os intervenientes acima mencionados, com o objetivo de conhecer os processos utilizados na UC de projeto-estágio da Licenciatura em Engenharia Eletrotécnica e de Computadores.

Assim, foram fornecidas as seguintes informações e sugestões:

- Tanto docentes como empresas externas, podem criar propostas de projeto-estágio
- Já existiu nesta UC, uma aplicação que permita a gestão das propostas. O docente referiu que era importante a existência de uma aplicação nesta tarefa, dado que assim, era possível os estudantes consultarem os projetos disponíveis e os diversos docentes verificarem os estados de cada um dos alunos (projeto atribuído, em processo de seleção, etc.).
- Os alunos indicam as preferências sobre as propostas existentes e contactam o propoente, sendo que é este que decide sobre o estudante que irá realizar o projeto.
- Não existem entregas parciais que os alunos devem cumprir, sendo que fica a cargo dos orientadores verificar se o progresso do aluno no desenvolvimento do projeto.
- Os docentes são convocados a arguir as provas finais, sendo que o RUC apenas valida a sua disponibilidade aquando da marcação das mesmas. Não existe um processo de convite de arguentes.
- O júri da prova é constituído por 4 elementos: orientador, presidente e 2 arguentes, sendo que cada um deles tem um peso na nota final atribuída.
- O RUC considera mais importante o desenvolvimento de uma plataforma que permita a gestão das propostas do que um software que permita auxiliar a marcação e reunião de júri para as provas.

- O RUC considera que é de extrema importância a plataforma a desenvolver, enviar vários alertas, nomeadamente via email (para o proponente) do estado das propostas.

Anexo C – Descrição dos requisitos funcionais

- UC1 – Criar proposta
    - Este caso de uso tem como objetivo a criação de propostas no sistema. Desta forma, o proponente deve preencher o formulário de criação e aguardar a decisão.
- UC2 – Editar proposta
    - Caso o proponente assim pretenda pode alterar as informações submetidas na criação de proposta.
- UC3 – Eliminar proposta (Proponente)
    - Caso a proposta não seja aceite ou o proponente não queira continuar o processo, pode optar por eliminar a proposta.
- UC4 – Consultar estado da proposta
    - Tanto o proponente como os estudantes interessados podem consultar o estado em que as propostas se encontram.
- UC5 – Rever proposta
    - Dado que existem propostas de projeto que são submetidas por organizações externas ao ISEP, é necessário revê-las. Esta revisão pode consistir em sugestões de melhoria da definição da proposta ou na sua aceitação.
- UC6 – Atribuir proposta a estudante
    - Este caso de uso pode ser realizado pelo RUC ou proponente da proposta de projeto, conforme a UC que se encontra configurada. Estes têm a possibilidade de determinar qual o estudante que vai realizar um determinado projeto, tendo por base o processo de seleção em vigor (e.g. seriação por média).
- UC7 – Atribuir proposta a orientador
    - O RUC deve reunir a informação sobre a orientação da proposta e, consequentemente deve submeter esta informação no sistema.
- UC8 – Preencher candidatura

- O estudante deve preencher o formulário de candidatura a uma determinada proposta de projeto. Esta candidatura pode ser uma simples indicação de preferências ou o preenchimento de um questionário, que visa a posterior seleção do aluno.

- UC9 – Gerir formalização

    - O estudante tem a responsabilidade de gerir a formalização da proposta, deve para isso submeter no sistema todas as informações necessárias, para que o revisor possa avaliar o projeto a ser desenvolvido.

- UC10 – Publicar revisões de formalização

    - O RUC tem a responsabilidade de publicar as revisões realizadas, isto é enviar (e.g. via email) ao orientador e/ou estudante os resultados da correção que foi realizada à formalização entregue.

- UC11 – Rever formalização

    - Em algumas UC, a formalização do projeto é alvo de revisão. Desta forma, o revisor selecionado, deve rever a informação inserida pelo estudante e criar anotações, de forma a fornecer sugestões de melhoria ou correções.

- UC12 – Escolher orientador

    - Em algumas UC, o estudante tem como responsabilidade escolher um docente do ISEP para o orientar no projeto. Por outro lado, existem UC, em que é o RUC que seleciona o orientador mediante o tema do projeto.

- UC13 – Aceitar orientação

    - Um docente do ISEP tem a possibilidade de aceitar ou rejeitar orientar um determinado projeto. Este caso de uso permite que a partir do momento em que é espoletado, o docente seja associado ao projeto e assim, se torne o orientador do mesmo.

- UC14 – Rever relatório de projeto

    - Este caso de uso pode ser realizado por diferentes atores do sistema, sendo eles: o orientador e o arguente. Estes realizam a revisão em momentos diferentes do desenvolvimento do projeto. Enquanto, que o orientador acompanha a escrita do relatório/tese/dissertação, o arguente apenas tem acesso a este nos momentos de avaliação.

- UC15 – Importar dados de submissão no Portal

- A prova final deve ser submetida no Portal do ISEP, de forma a cumprir a legislação aplicável. Por isso, é necessário que o RUC da UC de projeto importe estas informações, para que o *software* esteja apto à realização das etapas seguintes (Marcação de Provas).

- UC16 – Rastrear entregas intermédias

    - Com base na UC de projeto configurada, esta rastreabilidade pode ser realizada pelo orientador ou pelo RUC. Se for o RUC, devem ser importadas as entregas intermédias recebidas no *Moodle* do ISEP. Mas, por outro lado, se for o orientador a realizar este processo, devem ser registadas as informações acerca do projeto, do estudante e das apreciações dadas pelo docente.

- UC17 – Marcar prova

    - O RUC deve reunir as informações necessárias sobre a constituição do júri de prova e proceder à respetiva marcação (calendarização e localização) no Portal.

- UC18 – Notificar membros do júri

    - Em algumas UC, após definir o júri da prova, o RUC deve notificar todos os elementos (e.g. via email).

- UC19 – Definir arguentes de prova

    - A definição de arguentes de prova é realizada pelo RUC. Este, após reunir as sugestões dadas pelo orientador e tomar a sua decisão, define quem serão os arguentes da prova.

- UC20 – Convidar membros do júri de prova

    - Em algumas UC de projeto, o RUC é responsável por convidar os membros do júri de prova. Assim, este deve enviar convites a todos os membros que pretende que estejam presentes na avaliação de prova, com vista à reunião das informações relacionadas com a disponibilidade de cada um.

- UC21 – Propor arguentes de prova

    - O orientador do projeto pode indicar um determinado número de sugestões de arguentes (~2 arguentes internos e ~2 arguentes externos).

- UC22 – Aceitar ser presidente de prova

    - O docente convidado pelo RUC pode aceitar ou rejeitar esta responsabilidade. Este caso de uso permite que a partir do momento em que é espoletado o docente seja associado à prova, tornando-se o presidente do júri desta.

- UC23 – Participar na marcação de prova

    - Os diversos intervenientes no processo de marcação de prova devem ter uma participação ativa no mesmo. Desta forma, poderão ser evitados problemas de compatibilidade de horários e assim tornar o processo mais rápido.

- UC24 – Aceitar arguição

    - Após receber um convite de arguição de uma prova, o potencial arguente pode aceitar ou recusar. Esta decisão deve ser comunicada ao sistema, como forma de associar o docente à prova e este se tornar o arguente da mesma. Desta forma, será possível que este seja notificado das etapas seguintes, nomeadamente a marcação da prova.

- UC25 – Aceitar apresentação em prova de projeto orientado

    - Apesar da marcação de prova ser realizada pelo RUC, os diversos intervenientes neste processo podem dar o seu parecer sobre a data e hora de marcação da prova. Estas sugestões devem ser tidas em conta pelo RUC aquando da marcação final da prova.

- UC26 – Preencher *outcomes*

    - No ato de avaliação da prova de projeto, o orientador e o arguente devem preencher os *outcomes* que o estudante atingiu.

- UC27 – Preencher ata de prova

    - O presidente de prova é responsável por realizar o preenchimento da ata. Este preenchimento deve ter em consideração, o preenchimento dos *outcomes* realizado no caso de uso “UC26 - Preencher *outcomes*”.

Anexo D – Benefícios/Sacrifícios da análise de valor para o cliente

O valor percecionado pode ser entendido como a relação entre o custo e o valor dado pelo cliente a um determinado produto (Zeithaml, 1988).

A Tabela 15 representa os *drivers* do valor percecionado que devem ser tidos em conta, aquando da definição dos benefícios e sacrifícios da análise de valor para o cliente. Estes *drivers* funcionam como padrões que devem ser considerados para que esta análise seja realizada corretamente.

Tabela 15 – Benefícios e Sacrifício (Woodall, 2003)

Benefits		**Sacrifices**
Attributes	**Outcomes**	
Perceived quality	Functional benefits	Price
Product quality	Utility	Market price
Quality	Use function	Monetary costs
Service quality	Aesthetic function	Financial
Technical quality	Operational benefits	Costs
Functional quality	Economy	Costs of use
Performance quality	Logistical benefits	Perceived costs
Service performance	Product benefits	Search costs
Service	Strategic benefits	Acquisition costs
Service support	Financial benefits	Opportunity costs
Special service aspects	Results for the costumer	Delivery and installation costs
Additional services	Social benefits	Costs of repair
Core solution	Security	Training and maintenance costs
Customization	Convenience	Non-monetary costs
Reliability	Enjoyment	Non-financial costs
Product characteristics	Appreciation from users	Relationship costs
Product attributes	Knowledge, humor	Psychological costs
Features	Self-expression	Time
Performance	Personal benefits	Human energy
	Association with social groups	Effort
	Affective arousal	

Assim, no âmbito do produto alvo desta análise de valor, como já mencionado, os RUC das UC de projeto-estágio têm diferentes apreciações e sugestões acerca do *software* a desenvolver, dado que estes têm especial atenção, com determinadas etapas do processo de gestão.

A Tabela 16 representa os benefícios e sacrifícios que a análise de valor tem para o cliente.

Tabela 16 – Valor percecionado

	Produto	**Serviço**	**Relação**
Benefício	- Redução dos processos manuais que existem atualmente; - Possibilidade de configuração do processo de gestão; - Capacidade de criar novos processos de gestão.	- Assistência na implantação e configuração do software.	- Melhoria da imagem dos cursos do ISEP; - Redução do tempo gasto no processo de gestão. - Redução do esforço cognitivo dos processos de gestão
Sacrifício	- Incapacidade de interligação com determinados produtos existentes - Incapacidade ou dificuldade em adotar o processo desejado por falta de suporte	- Custos de manutenção	- Tempo de configuração personalizada do software. - Aprendizagem da utilização do software.

Anexo E - Análise SWOT

A análise SWOT é uma ferramenta que permite analisar a organização em que o produto se irá inserir, tanto externa como internamente. A SWOT é dividida em forças (*Strenghts*), fraquezas (*Weaknesses*), oportunidades (*Opportunities*) e ameaças (*Threats*). Portanto, esta análise será aplicada ao ISEP, uma vez que é nesta organização que o *software* produzido irá ser aplicado, mas com especial atenção para as UC de projeto-estágio (Figura 76).

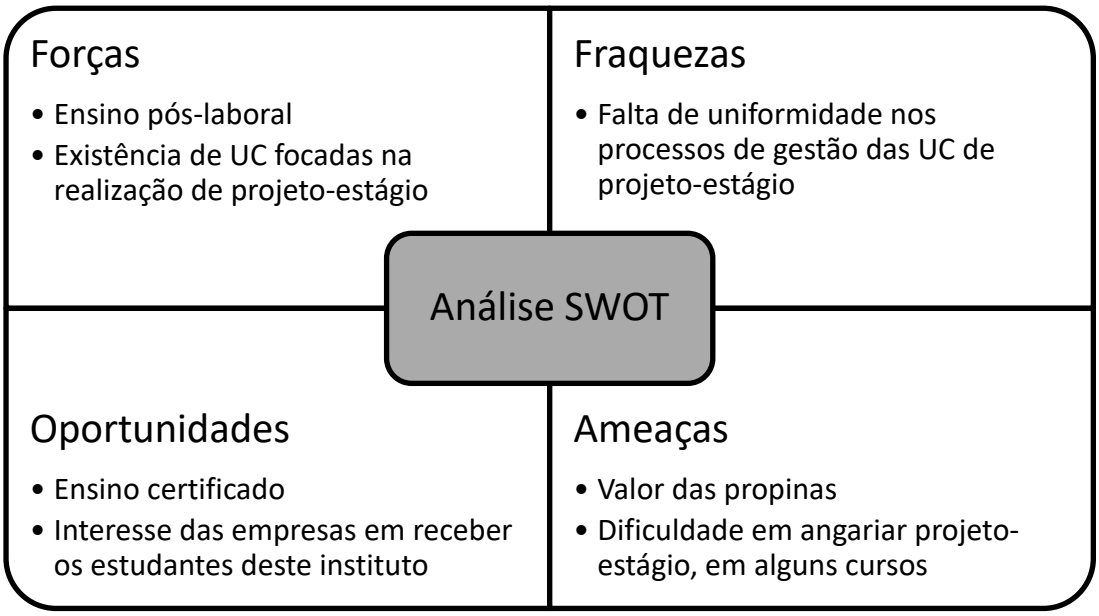

Figura 76 – Análise SWOT do ISEP

Anexo F – Método de Análise Hierárquica – AHP

Tendo em mente a proposta de valor enunciada neste documento (desenvolvimento de um *software* de apoio à gestão integrada de projetos-estágios das UC do ISEP), foi construída a árvore hierárquica de decisão, considerando as alternativas e critérios enunciados (Secção 5.1.2).

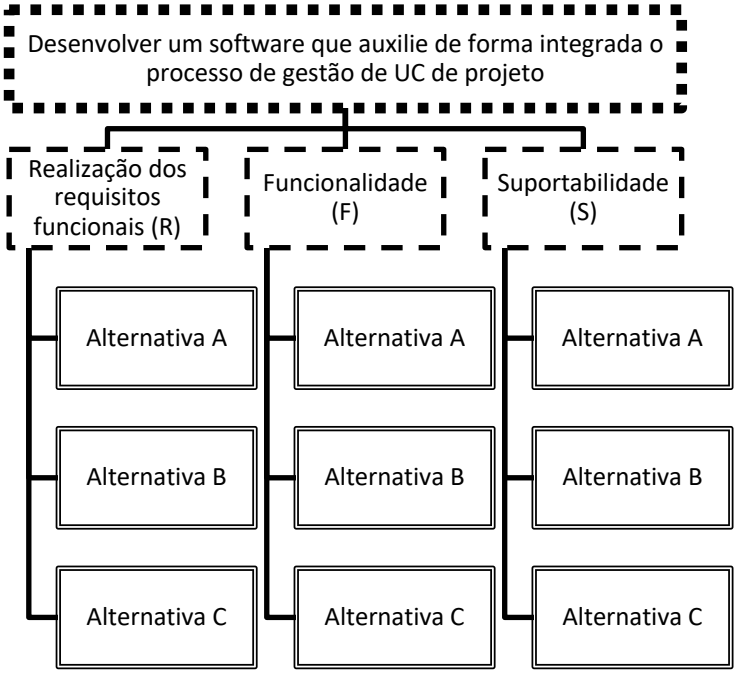

Figura 77 – Árvore hierárquica de decisão

A Figura 77 representa a árvore hierárquica de decisão, onde na base (contorno por pontos) é apresentado o problema, já anteriormente definido. As alternativas (contorno linha dupla) criadas, com base nos requisitos enunciados pelos RUC das UC e das assunções formuladas para o desenvolvimento do software (cf. Secção 5.1.3). Os critérios (contorno linha tracejada) representam as variáveis que serão avaliadas em cada uma das alternativas (cf. Secção 5.1.3).

De forma, a comparar os critérios definidos anteriormente é necessário atribuir-lhes níveis de importância. Estes níveis estão definidos na Escala Fundamental de Saaty (Tabela 17) (Saaty, 2008).

Tabela 17 – Escala Fundamental de Saaty

Nível de Importância	**Definição**	**Explicação**
1	Igual importância	As duas atividades contribuem igualmente para o objetivo.
3	Fraca importância	A experiência e o julgamento favorecem levemente uma atividade em relação à outra.
5	Forte importância	A experiência e o julgamento favorecem fortemente uma atividade em relação à outra.
7	Muito forte importância	Uma atividade é muito fortemente favorecida em relação a outra.
9	Importância absoluta	A evidência favorece uma atividade em relação a outra com o mais alto grau de certeza.
2,4,6,8	Valores intermediários	Quando se procura uma condição de compromisso entre duas definições.

A Tabela 18 representa as prioridades atribuídas a cada um dos critérios, tendo por base a Escala Fundamental de Saaty.

Tabela 18 – Matriz de comparação dos critérios do segundo nível

	R	F	S
R	1	1/8	1/7
F	8	1	3
S	7	1/3	1
Soma	16	1 1/2	4,14

Após o preenchimento da matriz de comparação é necessário proceder à soma de cada uma das colunas, com vista à obtenção da matriz normalizada dos critérios (Tabela 19).

Tabela 19 – Matriz normalizada dos critérios do segundo nível

	R	F	S
R	0,0625	0,0857	0,0345
F	0,5000	0,6857	0,7241
S	0,4375	0,2286	0,2414

Após a normalização da matriz de comparação é possível obter a importância de cada um dos critérios. Este vetor de prioridades é obtido através de média aritmética da matriz normalizada (Tabela 20).

Tabela 20 – Peso dos critérios

Critério de avaliação	**Peso**
Realização dos requisitos funcionais mais importantes para o cliente (R)	0,0609
Funcionalidade (F)	0,6366
Suportabilidade (S)	0,3025

Assim, já é possível concluir que o critério mais importante é a funcionalidade, seguida da suportabilidade. Para que seja possível decidir sobre a melhor ideia, é necessário verificar se a avaliação efetuada é consistente. Para tal, foi calculado o valor próprio (λmax):

$$Ax = \lambda_{máx}x \Leftrightarrow \begin{bmatrix} 1 & 1/8 & 1/7 \\ 8 & 1 & 3 \\ 7 & 1/3 & 1 \end{bmatrix} \times \begin{bmatrix} 0{,}0609 \\ 0{,}6366 \\ 0{,}3025 \end{bmatrix} = \lambda_{máx} \times \begin{bmatrix} 0{,}0609 \\ 0{,}6366 \\ 0{,}3025 \end{bmatrix}$$

$$\begin{bmatrix} 0{,}18368813 \\ 2{,}031260263 \\ 0{,}940982485 \end{bmatrix} = \lambda_{máx} \times \begin{bmatrix} 0{,}0609 \\ 0{,}6366 \\ 0{,}3025 \end{bmatrix} \Leftrightarrow \lambda_{máx} = 3{,}1059$$

Utilizando, este valor (3,1059) e sabendo que o número de critérios é 3, foi calculado o índice de consistência (IC):

$$IC = \frac{(\lambda_{máx} - n)}{(n-1)} \Leftrightarrow IC = \frac{(3{,}1059 - 3)}{(3-1)} \Leftrightarrow IC = 0{,}0530$$

O valor do índice aleatório (IR) é 0,58 (3 critérios) assim, obtém-se a razão de consistência (RC) através de:

$$RC = \frac{IC}{0{,}58} \Leftrightarrow RC = \frac{0{,}0530}{0{,}58} \Leftrightarrow RC = 0{,}0913$$

Uma vez que, é inferior a 0,1, podemos então concluir que os valores das prioridades relativas estão consistentes.

Anexo G - Resultados dos pedidos REST

- Exemplo de resposta ao pedido REST /schoolyear/1/edition/1/proposalconfig/1 /phaseconfig


```
{
    "_embedded": {
        "phaseConfigDTOList": [
            {
                "phaseConfigId": 2,
                "title": "Proposta",
                "titlePT": "Proposta",
                "titleEN": "Proposal",
                "owner": "swipetmdei@gmail.com",
                "sectionsConfig": [
                    1,
                    2,
                    3
                ],
                "_links": {
                    "changePhaseConfigByProposalConfigEditionSchoolYear": {
                        "href":
"https://uvmnps.dei.isep.ipp.pt/swipe/schoolyear/1/edition/1/proposalconfig
/1/phaseconfig/2"
                    }
                }
            },
            {
                "phaseConfigId": 1,
                "title": "Candidatura",
                "titlePT": "Candidatura",
                "titleEN": "Application",
                "owner": "swipetmdei@gmail.com",
                "sectionsConfig": [
                    4,
                    5,
                    6
                ],
                "_links": {
                    "changePhaseConfigByProposalConfigEditionSchoolYear": {
                        "href":
"https://uvmnps.dei.isep.ipp.pt/swipe/schoolyear/1/edition/1/proposalconfig
/1/phaseconfig/1"
                    }
                }
            }
        ]
    }
}
```


- Exemplo de resposta ao pedido REST /schoolyear/1/edition/1/proposalconfig/1 /phaseconfig/2/sectionconfig


```
[
    {
        "sectionConfigId": 3,
        "title": "Comunicação da decisão",
        "titlePT": "Comunicação da decisão",
```

```
            "titleEN": "Decision Comunication",
            "owner": "swipetmdei@gmail.com",
            "actionType": "SEND",
            "nextSectionConfigId": 4,
            "previousId": 1,
            "phasesConfigId": [
                2
            ],
            "fieldsConfig": [
                2
            ]
        },
        {
            "sectionConfigId": 2,
            "title": "Criação",
            "titlePT": "Criação",
            "titleEN": "Create",
            "owner": "swipetmdei@gmail.com",
            "actionType": "PUT",
            "nextSectionConfigId": 1,
            "previousId": -1,
            "phasesConfigId": [
                2
            ],
            "fieldsConfig": [
                4,
                7,
                8,
                11
            ]
        },
        {
            "sectionConfigId": 1,
            "title": "Decisão",
            "titlePT": "Decisão",
            "titleEN": "Decision",
            "owner": "swipetmdei@gmail.com",
            "actionType": "PUT",
            "nextSectionConfigId": 3,
            "previousId": 2,
            "phasesConfigId": [
                2
            ],
            "fieldsConfig": [
                1
            ]
        }
    ]
```


- Exemplo de resposta ao pedido REST /schoolyear/1/edition/1/proposalconfig/1 /phaseconfig/2/sectionconfig/2/fieldconfig


```
    [
        {
            "fieldConfigId": 8,
            "title": "Problema",
            "typeName": "Text",
            "owner": "swipetmdei@gmail.com",
            "required": true,
```

```
        "authorizations":
"Admin,RucInEdition,StudentInEdition,StudentInEntity,External,ProponentInEn
tity",
        "paramsConfig": [
            {
                "paramId": 8,
                "name": "qtdCaracteres",
                "description": "qtdCaracteres",
                "value": 500
            },
            {
                "paramId": 10,
                "name": "nLinhas",
                "description": "nLinhas",
                "value": 20
            }
        ],
        "languagesConfig": [
            {
                "languageId": 8,
                "name": "Problema",
                "description": "Descrição do problema a resolver",
                "code": "pt-PT"
            }
        ]
    },
    {
        "fieldConfigId": 11,
        "title": "Objectivos",
        "typeName": "Text",
        "owner": "swipetmdei@gmail.com",
        "required": true,
        "authorizations":
"Admin,RucInEdition,StudentInEdition,StudentInEntity,External,ProponentInEn
tity",
        "paramsConfig": [
            {
                "paramId": 13,
                "name": "qtdCaracteres",
                "description": "qtdCaracteres",
                "value": 500
            },
            {
                "paramId": 14,
                "name": "nLinhas",
                "description": "nLinhas",
                "value": 20
            }
        ],
        "languagesConfig": [
            {
                "languageId": 11,
                "name": "Objectivos",
                "description": "Lista de objectivos do projeto",
                "code": "pt-PT"
            }
        ]
    },
    {
        "fieldConfigId": 4,
```

```
        "title": "Organização",
        "typeName": "LineText",
        "owner": "swipetmdei@gmail.com",
        "required": true,
        "authorizations":
"Admin,RucInEdition,StudentInEdition,StudentInEntity,External,ProponentInEn
tity",
        "paramsConfig": [
            {
                "paramId": 4,
                "name": "qtdCaracteres",
                "description": "qtdCaracteres",
                "value": 50
            }
        ],
        "languagesConfig": [
            {
                "languageId": 4,
                "name": "Organização",
                "description": "Nome da organização",
                "code": "pt-PT"
            }
        ]
    },
    {
        "fieldConfigId": 7,
        "title": "Abordagem",
        "typeName": "Text",
        "owner": "swipetmdei@gmail.com",
        "required": true,
        "authorizations":
"Admin,RucInEdition,StudentInEdition,StudentInEntity,External,ProponentInEn
tity",
        "paramsConfig": [
            {
                "paramId": 7,
                "name": "qtdCaracteres",
                "description": "qtdCaracteres",
                "value": 500
            },
            {
                "paramId": 9,
                "name": "nLinhas",
                "description": "nLinhas",
                "value": 20
            }
        ],
        "languagesConfig": [
            {
                "languageId": 7,
                "name": "Abordagem",
                "description": "Descrição da abordagem a seguir",
                "code": "pt-PT"
            }
        ]
    }
]
```


- /schoolyear/1/edition/1/proposal/2/phase/3

```
[
    {
        "phaseId": 3,
        "owner": "swipetmdeiexternal@hotmail.com",
        "sections": [
            7,
            8
        ],
        "entityId": 2,
        "phaseConfigId": 2
    }
]
```


- /schoolyear/1/edition/1/proposal/2/phase/3/section/8


```
[
    {
        "sectionId": 8,
        "owner": "swipetmdeiteacher@gmail.com",
        "fields": [
            16
        ],
        "phaseId": 3,
        "sectionConfigId": 1
    }
]
```


- /schoolyear/1/edition/1/proposal/2/phase/3/section/8/field/16


```
[
    {
        "fieldId": 16,
        "owner": "swipetmdeiteacher@gmail.com",
        "value": "Validada",
        "sectionId": 8,
        "fieldConfigId": 1
    }
]
```