

Universidade do Minho - Escola de Engenharia

Relatório do trabalho prático de Desenvolvimento de Sistemas de Software

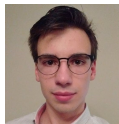
Sistema de Gestão de Turnos Práticos

Autores :

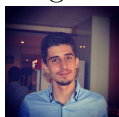
Diana Costa (A78985)



Marcos Pereira (A79116)



Sérgio Oliveira(A77730)



Vitor Castro(A77870)



Versão 1.0
11 de Novembro de 2017

Resumo

Neste relatório será feita uma abordagem inicial ao projeto de Desenvolvimento de Sistemas de Software ao qual está associado o desenvolvimento de um programa, em Java, responsável pela gestão dos turnos de um curso. Assim, este documento apresenta, detalhadamente, a perspectiva tomada pelo grupo em relação ao problema proposto pela equipa docente de DSS.

Conteúdo

1	Introdução	3
2	Problema	3
3	Solução	3
3.1	Esquema de Domínio	4
3.2	Esquema de Use Case	4
3.3	Especificação textual dos Use Case	5
3.3.1	Atribuir docente a turno	5
3.3.2	Atribuir turno a aluno	6
3.3.3	Atribuir UC a turno	6
3.3.4	Registrar docente	7
3.3.5	Criar turnos	7
3.3.6	Criar UC	7
3.3.7	Efectuar Login	8
3.3.8	Remover aluno de turno	8
3.3.9	Definir número máximo de alunos por turno prático	9
3.3.10	Inscribe aluno em turno	9
3.3.11	Efectuar Registo	10
3.3.12	Propôr Troca	10
3.4	Interface Gráfica	11
3.4.1	Login/Registo	11
3.4.2	Minha Área	12
3.4.3	Minhas Trocas	13
3.4.4	Ver Lista de Trocas Pendentes	14
4	Conclusões	15

1 Introdução

Este projeto tem como objetivo desenvolver um sistema capaz de alocar e gerir os turnos de um curso. A sua execução permitirá consolidar conhecimentos ao nível da programação em linguagens de objetos e introduz abordagens organizadas e estruturadas de desenvolvimento de software a partir de modelação e representação de dados em *UML 2.x*. Nesta fase intermédia do trabalho foi então sugerida a concepção da representação gráfica ou diagramas nessa linguagem.

2 Problema

Pretende-se desenvolver um sistema que atribui turnos (e, deste modo, um horário) aos alunos do curso e possibilita a ocorrência de trocas entre estes. As trocas estão condicionadas pela existência de dois alunos interessados em trocar de turno, no caso de pertencerem ao regime normal, e são feitas sem condicionalismos no caso do aluno ser trabalhador estudante. Neste momento, deve apresentar-se uma análise de requisitos da qual resultará um Modelo de Domínio, um Modelo de Use Case e uma proposta de interface gráfica.

3 Solução

A solução apresentada baseia-se na UML - Unified Modeling Language - linguagem útil na elaboração, modelação e documentação da estrutura de projetos de software e de sistemas orientados a objetos. Portanto, auxilia os *developers* de programas a visualizarem os seus sistemas através de diagramas padronizados. Até ao momento, a solução foi implementada com base em:

- Esquema de Domínio;
- Esquema de Use Case;
- Especificação dos Use Case;
- Proposta para a Interface Gráfica

3.1 Esquema de Domínio

O modelo de domínio analisa o problema de uma perspectiva concetual e é uma aproximação da representação gráfica das classes do programa e dos seus atributos, assim como o relacionamento entre estas.

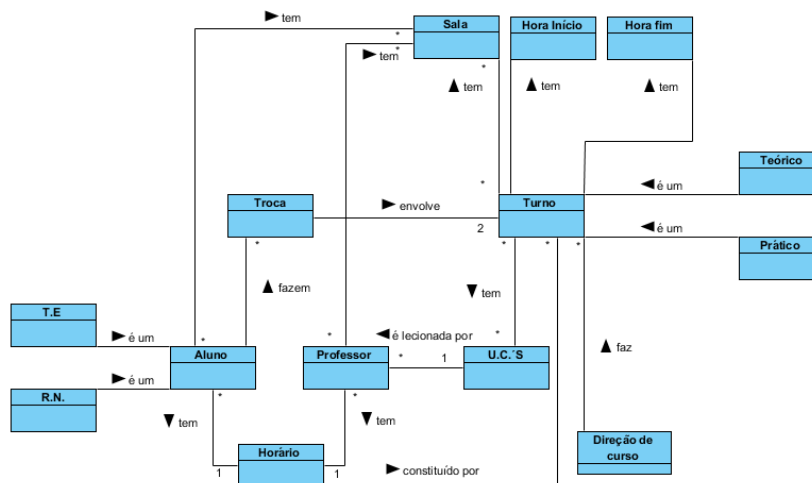


Figura 1: Modelo de Domínio proposto.

3.2 Esquema de Use Case

O esquema de Use Cases é o diagrama que mostra o que o programa faz do ponto de vista do utilizador, ou seja, consegue mostrar as principais funcionalidades do sistema e as suas interações com os atores do mesmo. Por isso, os diagramas de use case são compostos por atores (utilizadores do sistema), por use cases (funcionalidade) e pelas comunicações entre atores e use case. O grupo equacionou várias hipóteses para que fosse possível representar interações de utilizadores idênticos (como Trabalhador-estudante e Aluno, ou Docente-Responsável e Docente), e o resultado encontra-se na figura 2. Relativamente ao primeiro tuplo (Trabalhador-estudante e Aluno), decidiu-se pela unificação em Aluno, uma vez que o sistema será baseado numa bolsa de trocas, contrariamente ao plano inicial, que seria cada aluno propor a troca a um outro (tal como desenvolvido no sistema SWAP em vigor). Com esta bolsa de trocas, todos os alunos (trabalhadores-estudantes ou não) teriam igualdade de oportunidade de troca, tendo sido assumido que, caso houvesse um trabalhador-estudante com problemas de turnos, este contactaria diretamente a DC (tal como acontece na realidade), dispensando-se essa responsabilidade do programa. No que toca ao segundo tuplo (Docente-Responsável e Docente), depois de refletir sobre a pertinência de haver uma entidade Docente (que não responsável da UC) e sabendo que a única interação que poderia ter seria a de registar faltas, decidiu-se pela sua retirada. O motivo desta decisão prende-se pelo objetivo da Plataforma a ser desenvolvida, a saber, resolver um problema de atribuição de horários, e não pela marcação de faltas (apesar das mesmas terem uma implicação no horário de um Aluno, caso este falte a mais de 25 por cento das aulas). Deste modo, assume-se que as faltas serão contabilizadas tal como o são no momento e, a dada altura, o Docente-Responsável aperceber-se-á que determinado aluno deve ser removido do turno, funcionalidade essa que está disponível.

A Direção de Curso foi criada com o objetivo de poder ser o controlador principal de todo o sistema, sendo que pode fazer qualquer operação sem problema (mesmo que implique, a título de exemplo, inscrever um Aluno num turno que já está cheio). Tendo este caráter, qualquer problema

de compatibilidade operado por este interveniente não será da responsabilidade de verificação do programa.

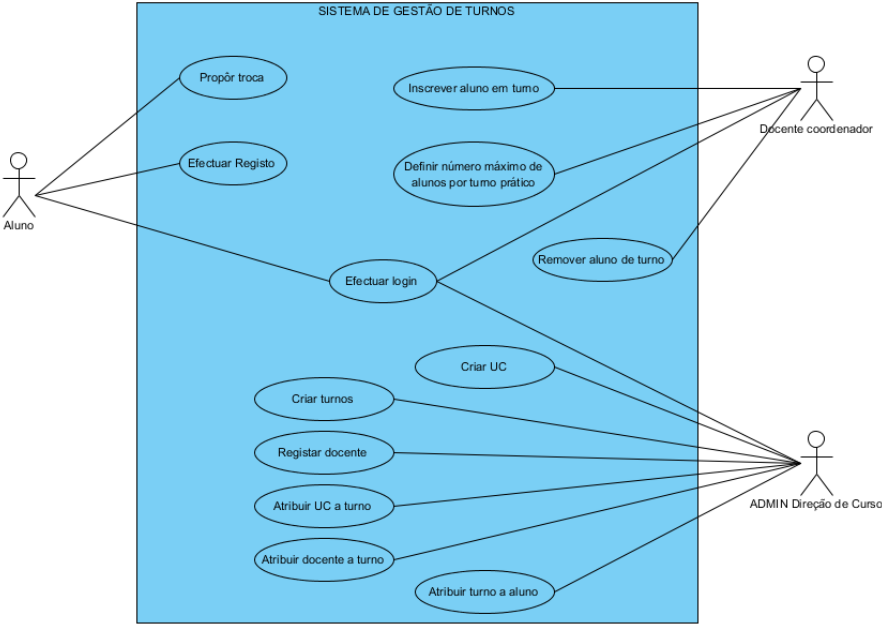


Figura 2: Diagrama de Use Cases proposto.

3.3 Especificação textual dos Use Case

Dado que um cenário é uma sequência de passos da interação entre ator e sistema, então a especificação textual dos casos de uso é um documento que descreve os vários cenários possíveis entre as comunicações de um mesmo objetivo ou funcionalidade. Apenas serão explicados alguns dos Use Cases (considerados mais importantes), uma vez que a sua explanação é suficiente para a compreensão de todo o problema.

3.3.1 Atribuir docente a turno

Brief Description			DC faz a atribuição de um docente a um turno
Preconditions			DC (logado) quer atribuir docente a turno
Post-conditions			Docente está atribuído a turno
Flow of Events		Actor Input	System Response
	1	Insere número de docente	
	2		Confirma existência de docente
	3		Confirma atribuição
Exceção 1 [docente não existe]		Actor Input	System Response
	1		Informa inexistência de docente

Figura 3

3.3.2 Atribuir turno a aluno

Brief Description	Diretor de curso atribui turno a aluno	
Preconditions	Diretor de curso (logado) quer atribuir turno a aluno	
Post-conditions	Aluno está inscrito em turno	
Flow of Events		Actor Input
	1	Insere número de aluno
	2	
	3	Seleciona UC e Turno a atribuir
	4	
		System Response
		Confirma número de aluno
		Regista mudança
Exceção 1 [número de aluno inválido]		Actor Input
	1	
		System Response
		Informa que número introduzido é inválido

Figura 4

3.3.3 Atribuir UC a turno

Responsabilidade do DC, que poderá atribuir a um turno a correspondente UC. O sistema apenas verifica a existência do turno e UC especificados, registrando a atribuição.

Brief Description	Turno alocado a uma UC	
Preconditions	UC pretende alocar um turno	
Post-conditions	Turno alocado	
Flow of Events		Actor Input
	1	Seleciona turno e UC
	2	
	3	
		System Response
		Verifica seleção
		Informa atribuição
Alternativa 1 [seleção inválida] (passo 2)		Actor Input
	1	
		System Response
		Informa seleção inválida

Figura 5

3.3.4 Registrar docente

Brief Description	Diretor de Curso cria docente	
Preconditions	DC logado quer criar docente	
Post-conditions	Docente criado	
Flow of Events		Actor Input
	1	Insere credenciais para o novo docente
	2	
	3	Seleciona responsabilidade
	4	
		System Response
		Confirma validade das credenciais
		Confirma criação de docente
Exceção 1 [credenciais erradas]		
		Actor Input
	1	
		System Response
		Informa credenciais não válidas

Figura 6

3.3.5 Criar turnos

Brief Description	Criação de um turno	
Preconditions	DC quer criar um turno	
Post-conditions	Turno criado	
Flow of Events		Actor Input
	1	Insere tipo, sala, lotação e escola
	2	
	3	
		System Response
		Verifica dados inseridos
		Regista turno
Exceção 1 [dados inseridos inválidos]		
		Actor Input
	1	
		System Response
		Sistema informa que dados inseridos são inválidos

Figura 7

3.3.6 Criar UC

Responsabilidade do DC, que deve inserir a UC no sistema, preenchendo os campos necessários à caracterização da mesma. O sistema verifica a inserção dos dados e cria a UC, caso tudo esteja correto.

Preconditions	DC quer criar Unidade Curricular	
Post-conditions	Unidade Curricular criada	
Flow of Events		Actor Input
	1	Insere ID, Ano, Descrição e ECTS
	2	
	3	
		System Response
		Confirma dados inseridos
		Regista UC
Brief Description	Criação de uma UC	
Exceção 1 [dados inválidos] (passo 2)		
		Actor Input
	1	
		System Response
		Informa dados inseridos inválidos

Figura 8

3.3.7 Efectuar Login

O utilizador (Aluno, Docente ou DC) faz login. As suas credenciais são verificadas e, caso aprovadas, terá acesso à sua área pessoal de acordo com os seus privilégios. O login é negado caso o utilizador seja inexistente ou a password não seja a correta.

Brief Description	Fazer o login no sistema (depois de registado)	
Preconditions	Ator não autenticado	
Post-conditions	Ator autenticado	
Flow of Events		Actor Input
		System Response
	1	Insere nº universitário e password
	2	Valida login e password
	3	Indica que o utilizador está autenticado
Exceção 1 [Credenciais inválidas] (Passo 2)		Actor Input
	1	Indica que os dados fornecidos são inválidos

Figura 9

3.3.8 Remover aluno de turno

Responsabilidade atribuída ao Docente-Responsável, que apenas terá de indicar o aluno que quer remover e. O sistema verifica a existência do aluno e será responsável pela identificação do turno onde aquele aluno está inscrito, removendo-o.

Brief Description	Docente remove aluno do turno	
Preconditions	Aluno excedeu os 25% de faltas aceites e docente quer removê-lo	
Post-conditions	Aluno foi removido do turno	
Flow of Events		Actor Input
		System Response
	1	Insere nº de aluno
	2	Valida nº de aluno
	3	Remove aluno do turno
Exceção 1 [número de aluno inválido] (passo 2)		Actor Input
	1	Informa nº aluno inválido

Figura 10

3.3.9 Definir número máximo de alunos por turno prático

Responsabilidade atribuída ao Docente-Responsável, que poderá definir um número de alunos máximo dentro dos intervalo 0 até CAPACIDADE DA SALA, sendo esta a única verificação feita pelo sistema. No fim, o número será definido.

Preconditions	Docente quer definir número máximo de alunos e está logado		
Post-conditions	Número de alunos máximo definido		
Flow of Events		Actor Input	System Response
	1	Seleciona turno	
	2		Verifica seleção
	3	Insere limite máximo de alunos	
	4		Verifica capacidade da sala do turno
	5		Regista informações relativas ao número máximo de alunos
Exceção 1 [seleção inválida] (passo 2)		Actor Input	System Response
	1		Informa seleção inválida
Exceção 2 [capacidade inferior ao limite proposto] (passo 4)		Actor Input	System Response
	1		Informa limite inválido

Figura 11

3.3.10 Inscreve aluno em turno

Tarefa atribuída ao Docente-Responsável que, segundo uma política de utilização responsável, não terá qualquer problema em adicionar qualquer aluno ao turno desejado (da UC que leciona), mesmo que aquele esteja cheio. Apenas se verifica a existência daquele aluno no sistema.

Brief Description	Docente responsável pretende inscrever aluno em turno		
Preconditions	Docente (logado) quer inscrever aluno		
Post-conditions	Aluno inscrito		
Flow of Events		Actor Input	System Response
	1	Insere número de aluno	
	2		Confirma existência de aluno
	3	Seleciona turno	
	4		Confirma seleção
	5		Informa inscrição concluída
Exceção 1 [aluno inexistente] (passo 2)		Actor Input	System Response
	1		Informa aluno inexistente

Figura 12

3.3.11 Efectuar Registo

Só efetuado por alunos que, durante o registo, escolherão o seu regime e também as UCs a que se querem inscrever. É feita a verificação de inserção dos dados.

Brief Description	Aluno regista-se e seleciona as cadeiras	
Preconditions	Aluno não registado quer registar-se no sistema	
Post-conditions	Aluno está registado no sistema	
Flow of Events		Actor Input
	1	Inserir número e password
	2	
	3	
	4	
	5	Seleciona as cadeiras desejadas
	6	
	7	Seleciona estatuto
	8	
	9	
Exceção 1 [número já registado] (passo 2)		Actor Input
	1	
Exceção 2 [password inválida] (passo 3)		Actor Input
	1	
Exceção 3 [inscrição inválida] (passo 6)		Actor Input
	1	
Exceção 4 [estatuto não introduzido] (passo 8)		Actor Input
	1	

Figura 13

3.3.12 Propôr Troca

Só efetuado por alunos sendo que, após seleção do turno e UC desejados, é lançada essa proposta para a bolsa de trocas. Posteriormente (e não sendo visível ao utilizador), o sistema verificará se existem propostas com reciprocidade, efetuando a troca automaticamente. Apenas é verificada a seleção de UC e turnos.


Brief Description	Aluno propõe troca de turno que fica pendente em bolsa de trocas	
Preconditions	Aluno logado quer trocar	
Post-conditions	Proposta de troca efetuada e a aguardar correspondência	
Flow of Events		Actor Input
	1	Seleciona a UC e turno desejados
	2	
	3	

Figura 14

3.4 Interface Gráfica

Segue-se a de prototipagem simples da interface, que poderá ser alterada subtilmente, conforme pequenos ajustes do programa a desenvolver.

3.4.1 Login/Registo



The logo features the text "GodSwap" in a bold, sans-serif font. To its right is a black silhouette of a person's head and shoulders. Overlaid on the silhouette are two white arrows: one pointing right and one pointing left, both originating from the center of the chest area.

Área de login:	Área de registo do aluno:
Número de ID: <input type="text"/>	Número de ID: <input type="text"/>
Palavra-passe: <input type="password"/>	Palavra-passe: <input type="password"/>
	Estatuto: <input checked="" type="radio"/> Trabalhador-Estudante <input type="radio"/> Regime Normal
	Uc's pretendidas: <input checked="" type="checkbox"/> Cadeira_ano1_sem1 <input type="checkbox"/> Cadeira_ano1_sem2 <input checked="" type="checkbox"/> Cadeira_ano2_sem1 <input checked="" type="checkbox"/> Cadeira_ano2_sem2 <input type="checkbox"/> Cadeira_ano3_sem1 <input type="checkbox"/> Cadeira_ano3_sem2


Figura 15: Página inicial da interface e área de registo.

3.4.2 Minha Área

Minha Área

Minhas Trocas

Ver Lista de Trocas Pendentes



Número de ID: xxxxxx

Estatuto: xxxxxx

Minhas UC's:

UC1: PL1 (9:00h - 11:00h)

UC2: PL2 (16:00h - 18:00h)

UC3: PL3 (9:00h - 11:00h)

UC4: PL4 (11:00h - 13:00h)

UC5: PL5 (9:00h - 11:00h)

UC6: PL6 (15:00h - 17:00h)

[Logout](#)

Figura 16: Área pessoal do utilizador em questão.

3.4.3 Minhas Trocas

Minha Área

Minhas Trocas

Ver Lista de Trocas Pendentes

Efetuar troca:

UC pretendida:

Turno pretendido:

Histórico de Trocas:

(dia-mes-ano):

- UC: xxx
- PLxx para PLyy
- Aluno envolvido: xxxxx

(dia-mes-ano):

- UC: xxx
- PLxx para PLyy
- Aluno envolvido: xxxxx

[Logout](#)

Figura 17: Histórico de trocas efetuadas e dados pessoais.

3.4.4 Ver Lista de Trocas Pendentes

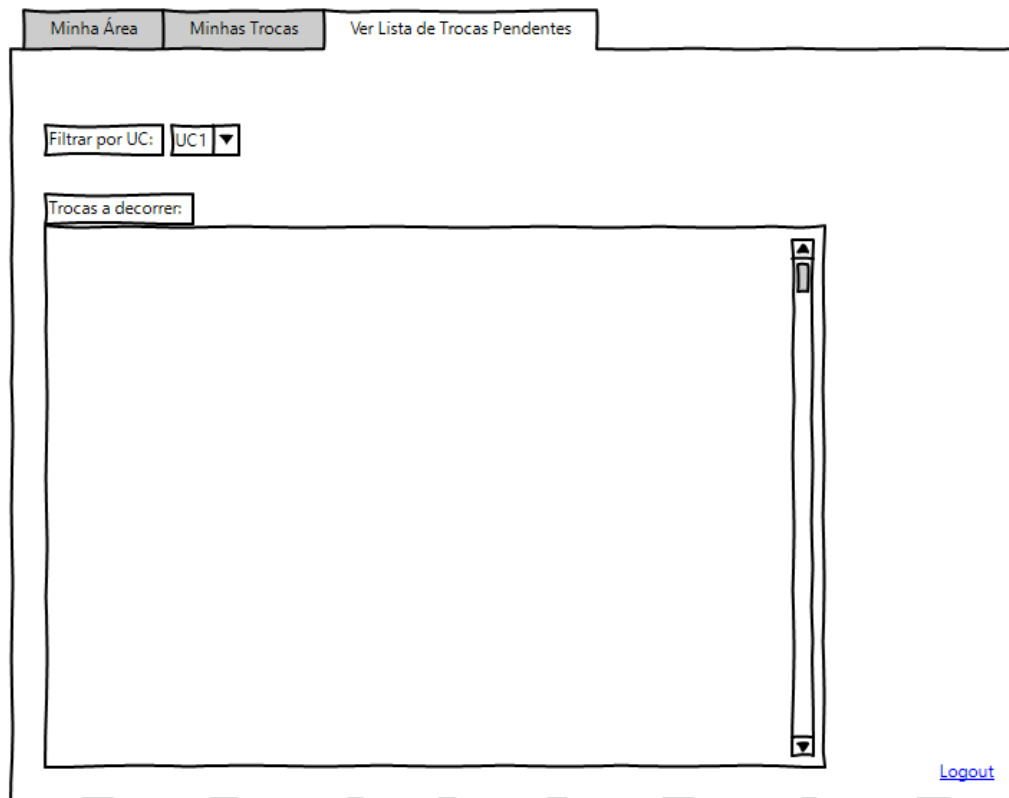


Figura 18: Lista de trocas requeridas por alunos e ainda pendentes.

4 Conclusões

Durante a realização desta parte inicial do projeto, foram encontradas algumas dificuldades, principalmente no que toca à realização do Diagrama de Use Cases. Depois de alguma discussão, a equipa definiu que tipo de implementação o sistema teria (i.e., se as trocas seriam propostas diretas ou em forma de bolsa(conjunto)) e, a partir desse momento, tornou-se mais fácil o processo de elaboração dos Use Cases e de definição dos Atores do sistema. Este processo foi importante pois permitiu a consciencialização das implicações que a escolha de determinados Use Cases teria no sistema, sendo que o grupo está confiante na elaboração do software correspondente, tendo consciência que outras dificuldades poderão surgir e que, eventualmente, faça sentido alterar o Diagrama.