
Relatório do Trabalho Prático

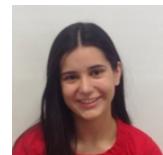
SISTEMA DE GESTÃO DE TURNOS

DESENVOLVIMENTO DE SISTEMAS DE SOFTWARE

Ana Catarina Lopes Carvalho Sousa A78029



Ana Sofia Gomes Marques A75248



Matias Nicolau Araújo A76234



Miguel Afonso Machado da Cunha A78478



Pedro Mendes Félix da Costa A79003



Conteúdo

1	Introdução	3
2	Modelo de Domínio	4
2.1	Descrição do Modelo de Domínio	4
2.2	Modelo de Domínio	6
3	Use Cases	7
3.1	Especificação dos Use Cases	8
3.1.1	Login	8
3.1.2	Pedir Troca	9
4	<i>Mockups</i>	10
5	Diagramas de Máquina Estado	11
5.1	Diagrama de Máquina de Estado Principal - Login	11
5.2	Diagrama de Máquina de Estado Aluno	12
5.3	Diagrama de Máquina de Estado Aluno com Estatuto Especial	13
5.4	Diagrama de Máquina de Estado Docente	14
5.5	Diagrama de Máquina de Estado Coordenador	15
5.6	Diagrama de Máquina de Estado Diretor de Curso	16
6	Diagrama de Package	17
7	Diagramas de Classes	18
8	Diagramas de Sequência com Subsistemas	21
8.1	Realizar Troca	21
8.2	Atribuir Turnos	23
9	Diagrama de Instalação	27
10	Base de Dados	28
11	Conclusão e trabalho futuro	29
A	Especificação dos Use Cases	30
A.1	Login	30
A.2	Escolher UCs	30
A.3	Consultar UCs em que está inscrito	31
A.4	Consultar os turnos que lhe foram atribuídos	31
A.5	Aceitar sugestão de troca	31
A.6	Mudar de turno	32

A.7	Marcar presenças	32
A.8	Consultar turnos que leciona	32
A.9	Adicionar aluno a um turno	33
A.10	Remover aluno a um turno	33
A.11	Importar UCs	34
A.12	Importar alunos	34
A.13	Importar turnos	35
A.14	Ativar logins	35
A.15	Apagar turno	36
A.16	Adicionar turno	36
A.17	Consultar UC	37
A.18	Consultar turnos	37
A.19	Atribuir Turnos	38
B	<i>Mockups</i>	39
B.1	Utilizador	39
B.2	Aluno	40
B.3	Aluno com estatuto especial	42
B.4	Docente	43
B.5	Coordenador	44
B.6	Diretor de Curso	45

1. Introdução

Este relatório visa apresentar as decisões tomadas durante a realização do trabalho prático da Unidade Curricular de Desenvolvimento de Sistemas de Software. Procuramos justificar todas as considerações feitas na formulação do problema e na elaboração dos modelos e diagramas que o suportam: Modelo de Domínio, Use Cases (e respetiva especificação), Diagramas de Máquinas de Estado, *mockups*, Diagrama de Classes, Diagrama de Sequência com Submissões, Diagrama de *Package* e Diagrama de Instalação.

Toda a modulação do problema, através de diagramas, esquemas e especificações foi realizada com recurso à linguagem de modelação UML, abordada nesta unidade curricular.

2. Modelo de Domínio

O sistema a implementar destina-se a suportar a configuração e gestão de turnos práticos de um determinado curso de licenciatura ou mestrado integrado.

Neste capítulo são apresentados os requisitos do problema e uma proposta de Modelo de Domínio.

2.1 Descrição do Modelo de Domínio

O problema proposto é desenvolver um sistema de gestão de turnos práticos. Após uma análise das necessidades e de factos reais, percebeu-se a existência dos seguintes conceitos, a integrar na modelação do problema: Aluno, Turno, UC (Unidade Curricular), Trocas (de turno(s) que o aluno pretende efetuar), Mudança (de turno) e Faltas.

Posto isto, é importante analisar como os conceitos recolhidos se relacionam entre si. Tem-se que:

- cada Aluno frequenta 1 ou várias UC's;
- cada UC possui 1 ou vários turnos;
- cada Turno é frequentado por vários alunos;
- cada Aluno pode frequentar 1 ou vários turnos, dependendo do número de UC's que frequenta, e se estas têm apenas aulas práticas/ práticas-laboratoriais ou se são constituídas por aulas práticas/práticas-laboratoriais e aulas teóricas. Por exemplo, se um aluno tem apenas uma Unidade Curricular, pode frequentar um ou dois turnos, dependendo das questões levantadas acima; se estiver inscrito a várias UC's frequenta vários turnos.

Os turnos em que os alunos são alocados são previamente definidos pelo Diretor de Curso e têm associados um determinado número de vagas, um ou vários dias específicos em que são lecionados e um determinado horário constituído por hora de início e hora de fim. Qualquer aluno pode, posteriormente, efetuar várias trocas de turno, desde que encontre outro aluno com quem efetuar essa troca.

Por forma a efetivar as trocas, é necessário que o coordenador execute a respetiva mudança de turno.

Associado a cada aluno, temos o seu email de aluno e o Nome, elementos que o identificam. Ainda no que diz respeito ao Aluno, este pode ter estatuto especial de Trabalhador Estudante e, apenas nesse caso, pode efetuar as mudanças de turno que desejar, sem ter a necessidade de trocar com outro aluno, desde que exista capacidade no turno que pretende.

A capacidade de um Turno depende da sala em que o é lecionado e do seu tipo; turnos práticos e práticos-laboratoriais têm limite máximo de alunos definido pelo coordenador da respetiva UC.

Existe ainda um sistema de faltas que pode ser definido através de um relacionamento ternário entre um docente que marca zero ou várias faltas, o aluno que comete as faltas, podendo estas ser zero ou várias, e ainda o turno a que as faltas são cometidas, podendo ser cometidas zero ou várias faltas a cada turno.

2.2 Modelo de Domínio

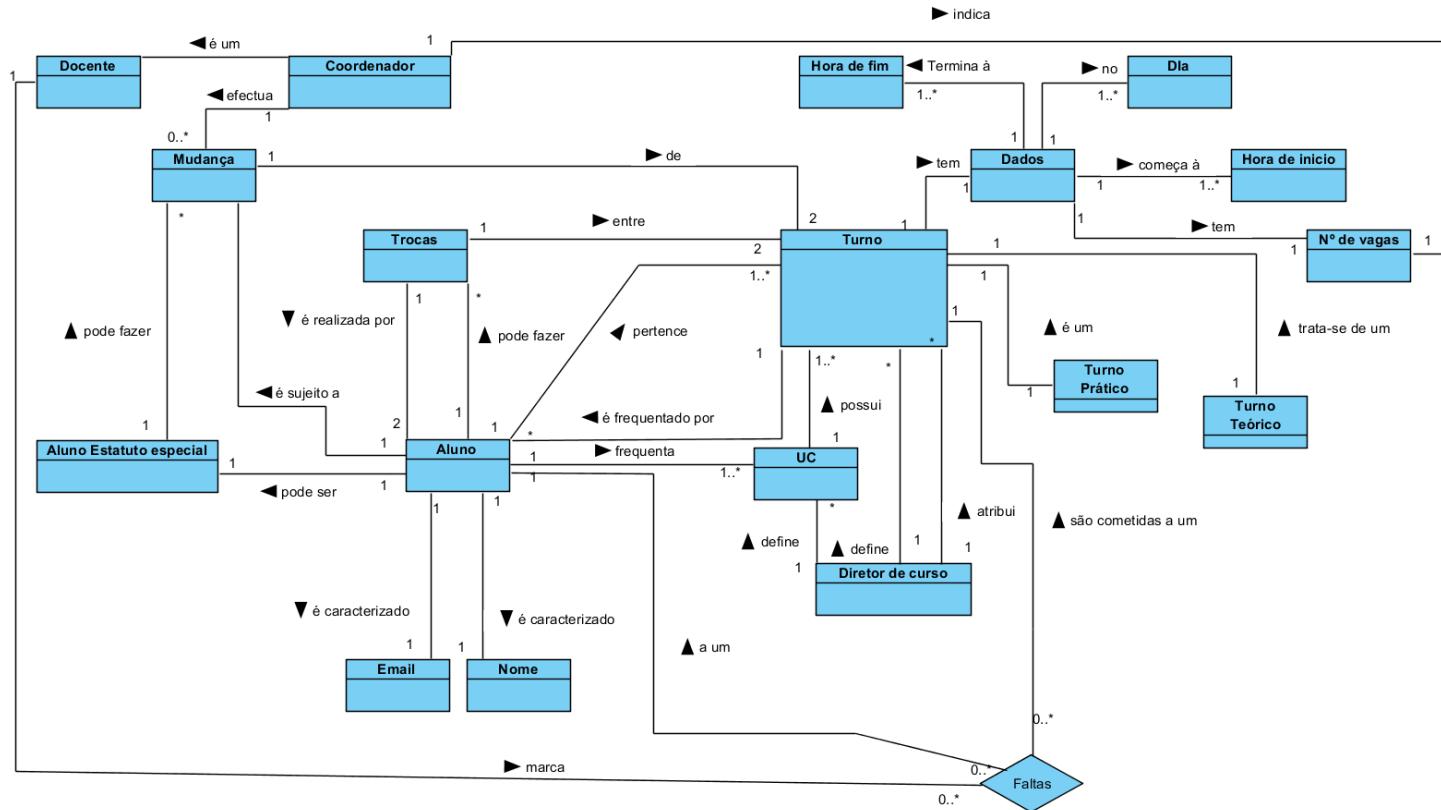


Figura 2.1: Modelo de Domínio

3. Use Cases

O Diagrama de Use Case representa os atores do sistema e as respetivas tarefas de cada um deles. Para os *Use Cases* foram considerados seis atores:

- **Utilizador:** Todos os outros atores são utilizadores que necessitam de fazer o login para se autenticarem.
- **Aluno:** Responsável por escolher as UC's que pretende frequentar e por solicitar o pedido troca de turno, assim como por aceitar sugestões de troca por parte de outros colegas. Pode ainda consultar as UC's e os turnos em que está inscrito.
- **Estatuto Especial:** É um Aluno que, para além das ações normais deste ator, pode também realizar trocas de turno sem precisar de trocar com outro colega.
- **Docente:** Responsável efetuar a marcação de presenças nos turnos que leciona. Pode também consultar as informações que dizem respeito aos turnos por ele lecionados.
- **Coordenador:** É o docente responsável pela UC e, portanto, é quem efetua as trocas de turnos, tanto para alunos normais, como para alunos com estatuto especial, adicionando e removendo o aluno a determinado turno.
- **Diretor de Curso:** Responsável por gerir as UC's, os turnos e os alunos que os frequentam.

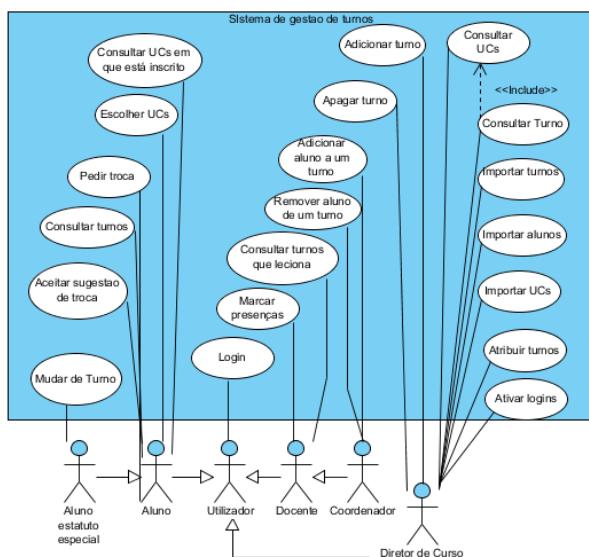


Figura 3.1: Diagrama de *Use Case*

3.1 Especificação dos Use Cases

3.1.1 Login

O use case Login pertence ao Utilizador e é o único *Use Case* que diz respeito a todos os atores do sistema e que consiste, basicamente, no processo de autenticação necessário para que se possa interagir com o sistema.

O utilizador insere as suas credenciais e o sistema procede à validação das mesmas. Se estas forem válidas, o utilizador fica autenticado, caso contrário, isto é, as credenciais são válidas (exceção 1), o sistema informa o utilizador de que uma (ou ambas) das credenciais que inseriu (email ou password) não é válida.

Use Case: Login		
Descrição: Utilizador efetua login		
Pré-condição: --		
Pós-condição: Utilizador fica autenticado.		
	Actor	Sistema
Comportamento Normal	1. Inserir credenciais	
		1. Valida credenciais
		2. Autentica o utilizador
		3. Informa de sucesso
Exceção 1 [Credenciais Inválidas] (passo 1)		1.1. Informa que o email ou password não são válidos

Figura 3.2: Especificação do *Use Case* Login

3.1.2 Pedir Troca

O use case Pedir Troca é efetuado pelo Aluno, que, para o poder fazer, tem de estar autenticado no sistema e tem de ter turnos atribuídos.

O Aluno começa por indicar qual o turno para que pretende ir. O sistema verifica se não existe nenhuma sobreposição com outros turnos e informa que o pedido foi registado com sucesso. Caso haja sobreposição (alternativa 1), o sistema avisa o aluno deste facto e permite que ele opte por prosseguir com o pedido ou por cancelá-lo (exceção 2). Há ainda a hipótese de o turno indicado pelo aluno não existir (exceção 1), pelo que, nesse caso, o sistema informa o Aluno que o turno escolhido não existe.

Use Case: Pedir troca		
Descrição: Utilizador pede para efetuar uma troca.		
Pré-condição: O Utilizador está autenticado como Aluno, tem turnos atribuídos e o período de aulas ainda não começou.		
Pós-condição: O Utilizador fica com um pedido de troca registado.		
	Actor	Sistema
Comportamento Normal	1. O Aluno indica o turno para que pretende ir.	
		2. Verifica se haverá sobreposição
		3. O sistema regista o pedido
		4. Informa que o pedido foi registado com sucesso.
Exceção 1 [Turno não existe] (passo 2)		2.1. Informa que o turno não existe
Alternativa 1 [Mudança implica sobreposição de turnos] (passo 2)		2.1. Informa com que turno vai ter sobreposição.
	2.2. Indica que quer pedir a troca na mesma.	2.3. Regressa ao passo 3.
Exceção 2 [Não quer pedir o turno] (passo 2.2)		2.2.1. Informa que o pedido foi cancelado.

Figura 3.3: Especificação do *Use Case* Pedir Troca

No que diz respeito à especificação dos *Use Case*, optamos por explicar apenas estes, como exemplo. As restantes especificações podem ser observadas no Anexo A.

4. *Mockups*

Os protótipos de interface, também conhecidos como *mockups* traduzem os requisitos do sistema, assim como toda a modelação até agora desenvolvida. Apresentamos um exemplo do *mockup* Login, que permite que qualquer utilizador se autentique, apresentando o seu número e a password, e clicando depois no botão Login. Caso os dados inseridos estejam corretos, este fica autenticado, caso contrário, será emitida uma mensagem de erro. Os restantes *mockups* podem ser consultados no Anexo B.

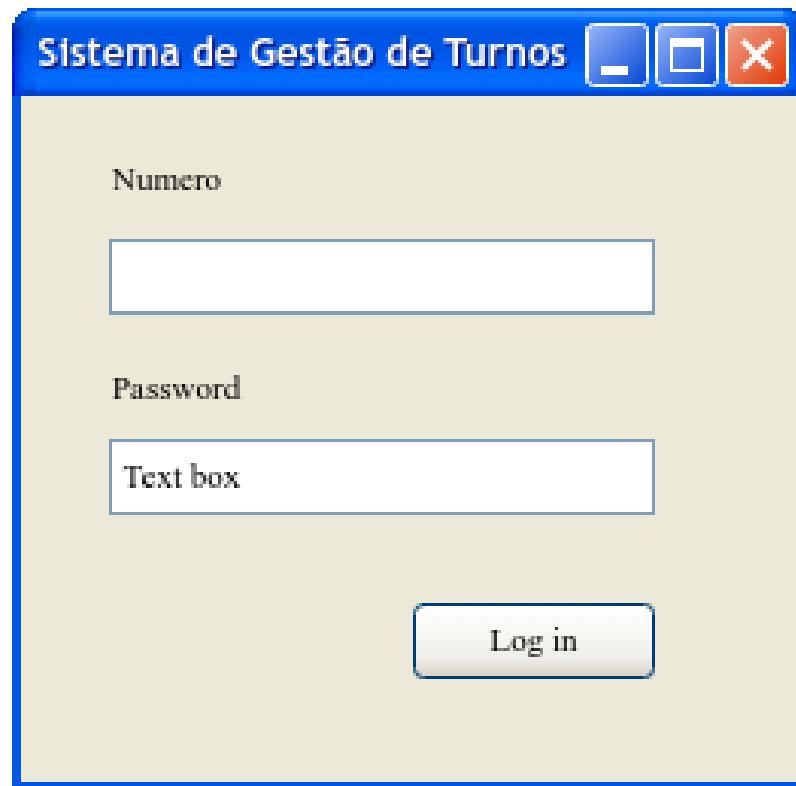


Figura 4.1: *Mockup Login*

5. Diagramas de Máquina Estado

A elaboração de diagramas de Máquinas de Estado permite modelar o comportamento do sistema como um todo. Estes diagramas são muito importantes, uma vez que permitem modelar todos os estados possíveis do sistema, em particular, os estados da interface gráfica. Para isto, foram modelados seis diagramas.

5.1 Diagrama de Máquina de Estado Principal - Login

O diagrama de máquina de estado principal *main* consiste no primeiro passo que cada utilizador tem de efetuar antes de poder interagir com o sistema. Quando tenta fazer Login, o utilizador tem de inserir as suas credenciais e, caso estas estejam corretas, acede ao estado correspondente ao seu perfil de utilizador. Sempre que pretender, pode efetuar logout, voltando para o estado *main*.

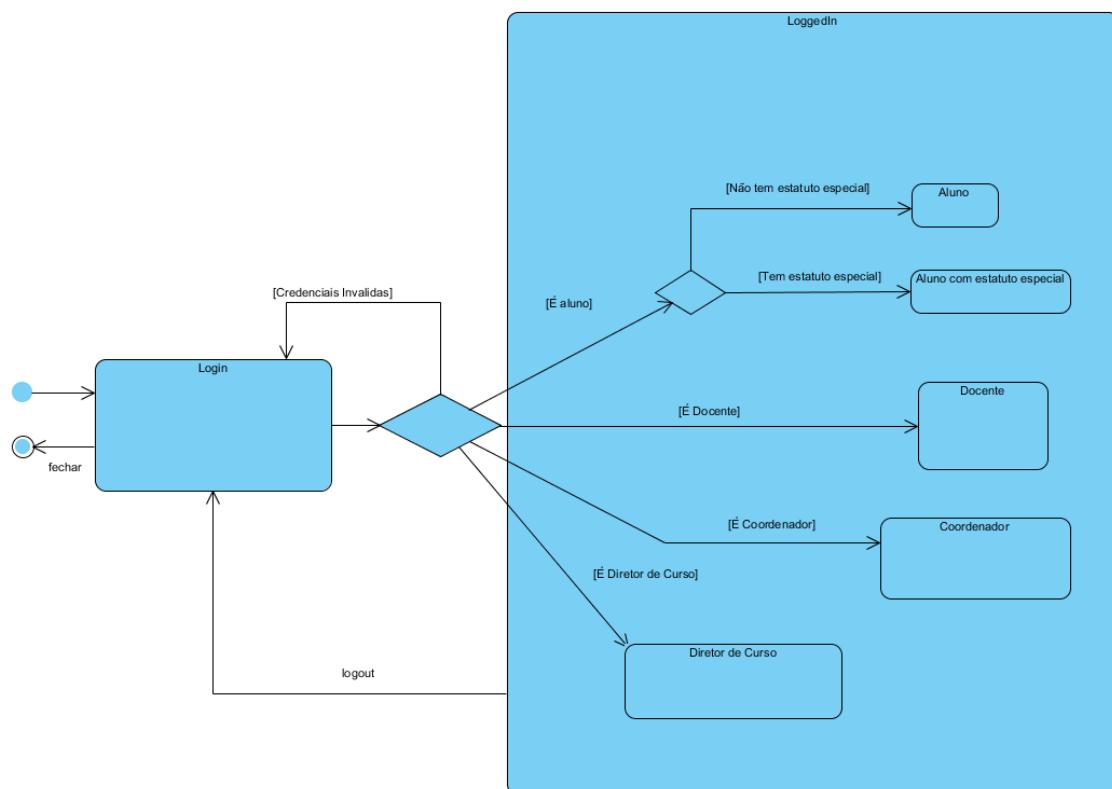


Figura 5.1: Diagrama de Máquina de Estado principal - Login

Existem ainda diagramas para cada um dos nossos atores:

5.2 Diagrama de Máquina de Estado Aluno

O aluno pode, dependendo do estado interno em que se encontra, alterar o seu estado de duas formas: se os seus turnos ainda não tiverem sido atribuídos, pode escolher as UCs que pretende frequentar. Por outro lado, se este já possuir turnos poderá registar ou pedidos de troca com outros alunos. No entanto, se os Diretor de curso proibir as trocas, este não pode alterar o seu estado para efetuar qualquer das operações anteriormente referidas.

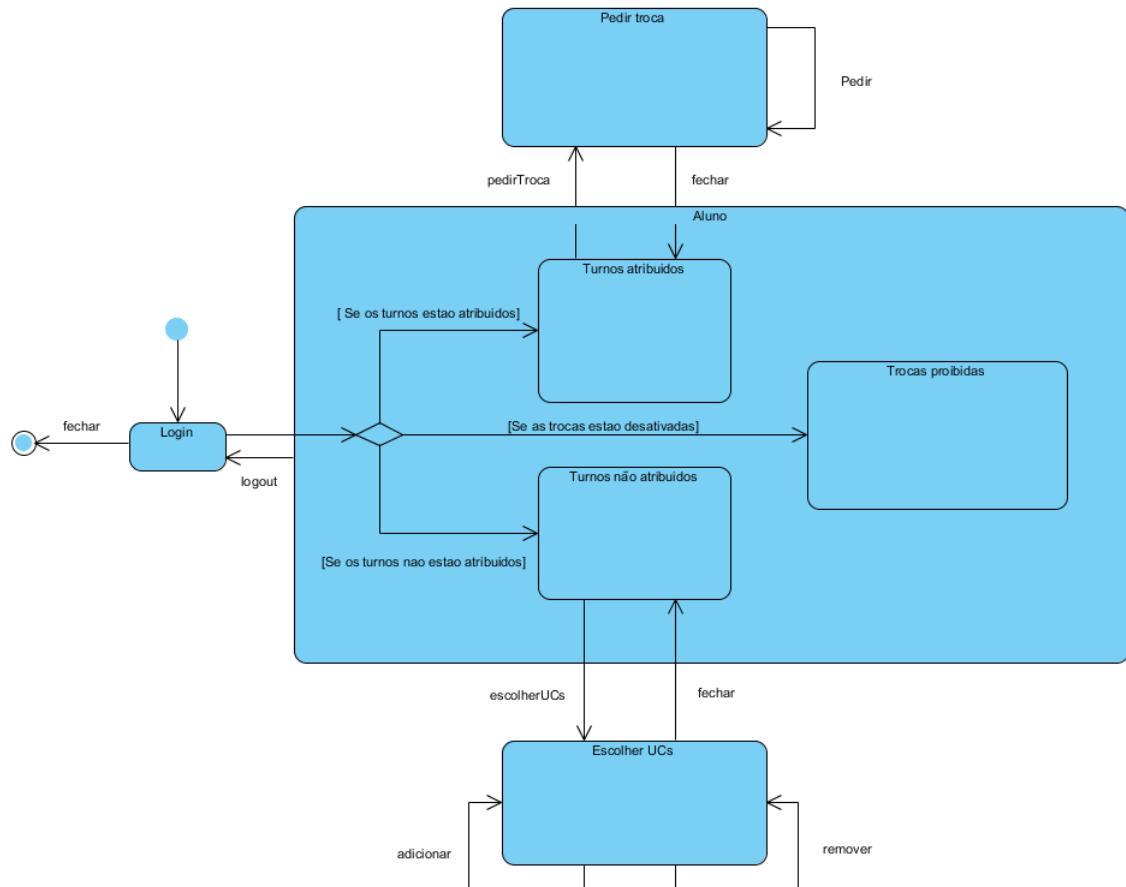


Figura 5.2: Diagrama de Máquina de Estado Aluno

5.3 Diagrama de Máquina de Estado Aluno com Estatuto Especial

O Aluno com Estatuto Especial é idêntico ao Aluno, excetuando o facto de poder mudar de turno sem ter de efetuar a troca com outro colega, ficando com o turno automaticamente atribuído. Esta operação só pode ser efetuada caso as trocas forem permitidas.

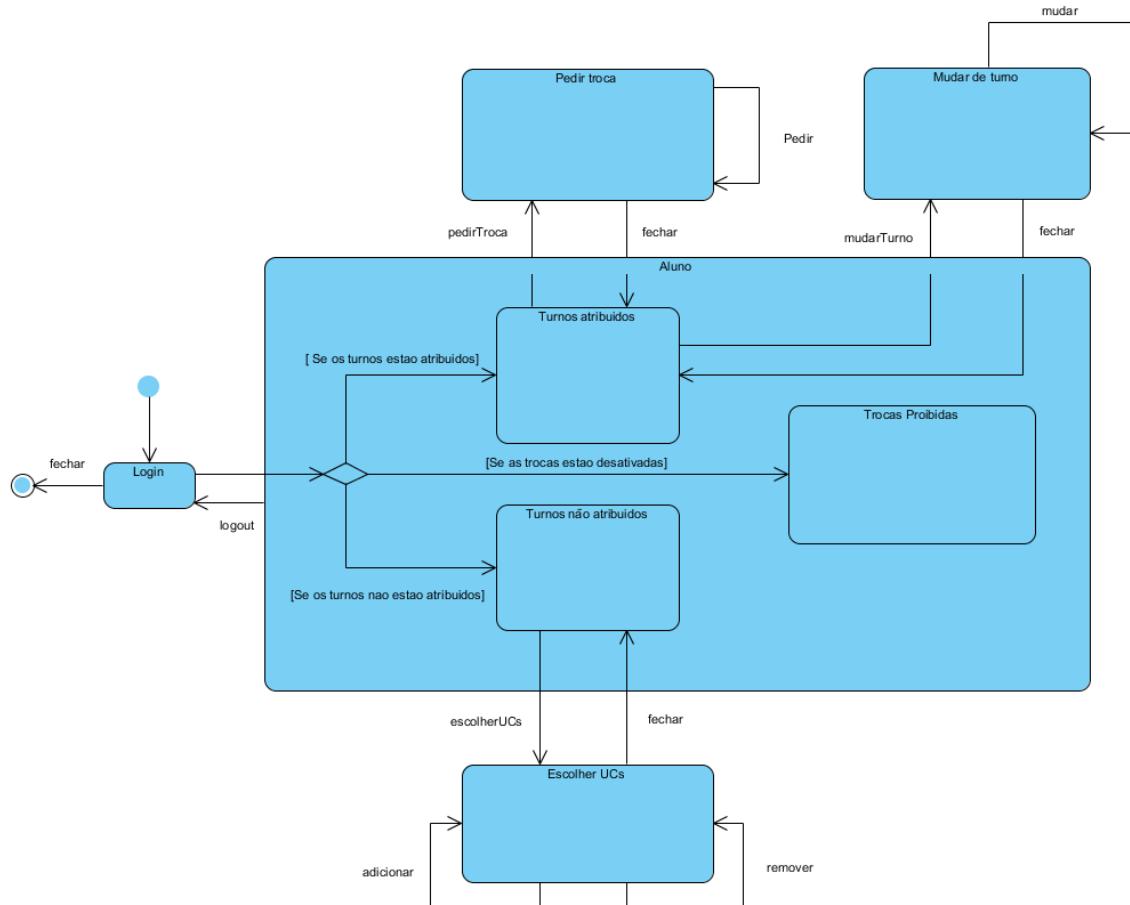


Figura 5.3: Diagrama de Máquina de Estado Aluno com Estatuto Especial

5.4 Diagrama de Máquina de Estado Docente

O Docente pode consultar as UC e os respetivos turnos que leciona. Pode ainda efetuar a marcação de presenças a esses turnos numa determinada aula.

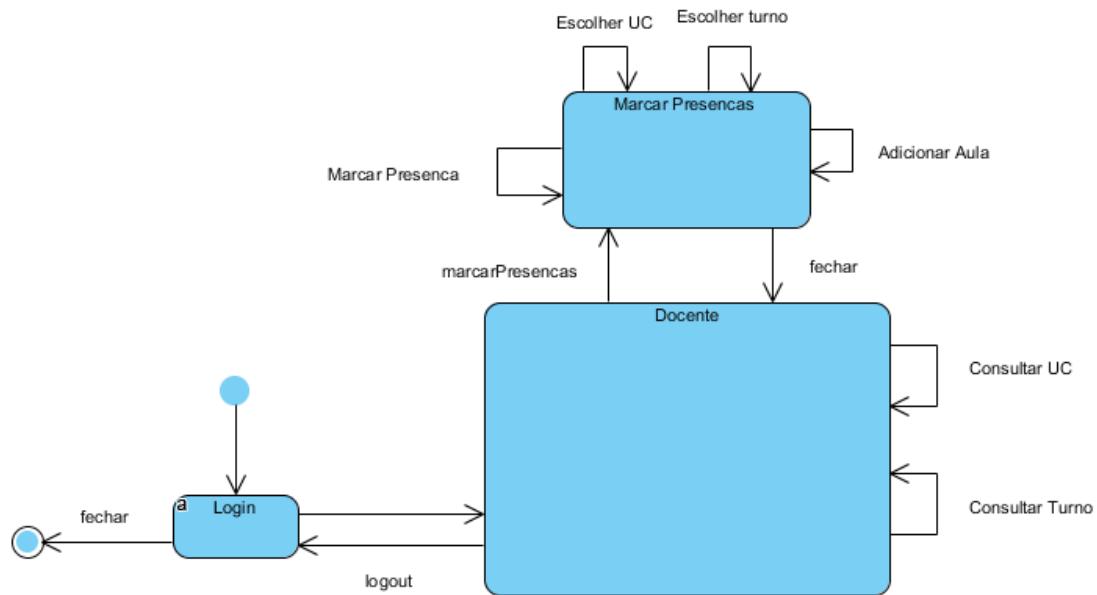


Figura 5.4: Diagrama de Máquina de Estado Docente

5.5 Diagrama de Máquina de Estado Coordenador

O Coordenador é em tudo semelhante ao Docente, acrescentando apenas o facto de este ser o responsável pelas trocas de turnos e assim poder adicionar e remover alunos aos turnos da UC que rege.

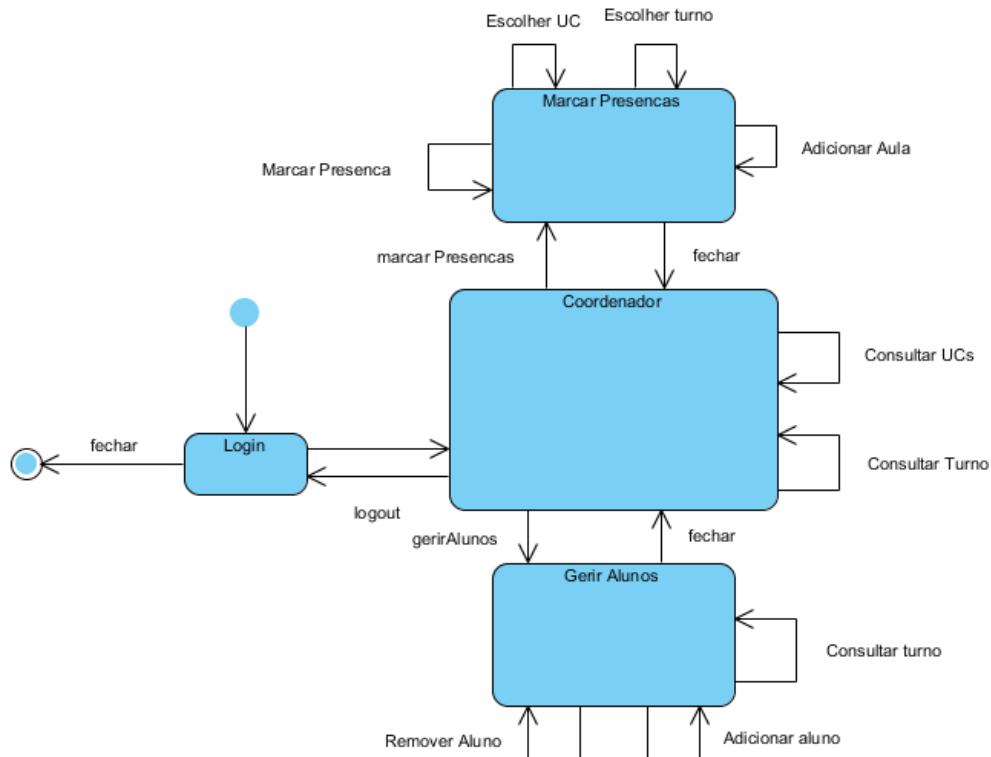


Figura 5.5: Diagrama de Máquina de Estado coordenador

5.6 Diagrama de Máquina de Estado Diretor de Curso

O Diretor de curso tem a responsabilidade de povoar o sistema com as UCs, os utilizadores e o turnos, ativar os logins do utilizadores, atribuir turnos aos utilizadores e de desativar as trocas. Todas estas operações tem de ser efetuadas numa determinada ordem e o diagrama de maquina de estado deste reflete esta necessidade, apenas podendo mudar para um determinado estado interno se todas a operações de que este depende tiverem sido efetuadas com sucesso.

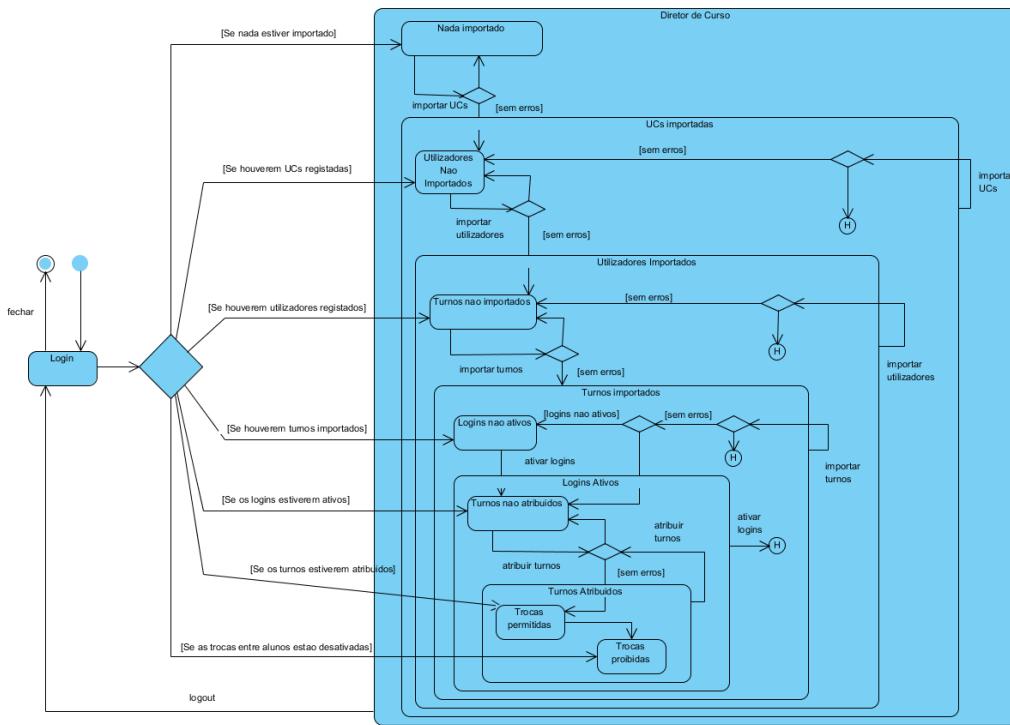


Figura 5.6: Diagrama de Máquina de Estado Diretor de Curso

6. Diagrama de Package

Existem três packages principais no nosso sistema:

- **userInterface** - implementa a interface gráfica com que o utilizador final vai interagir e faz uso do padrão de desenvolvimento *Observer*, *Observable* para o seu funcionamento.
- **sgt** - Contém a lógica de negócio da aplicação, controla todas as operações do sistema e implementa *Observable*;
- **dao** - Mapeia os objetos de uma base de dados relacional em MySQL para classes em Java.

O diagrama seguinte representa então esses packages e as relações entre eles.

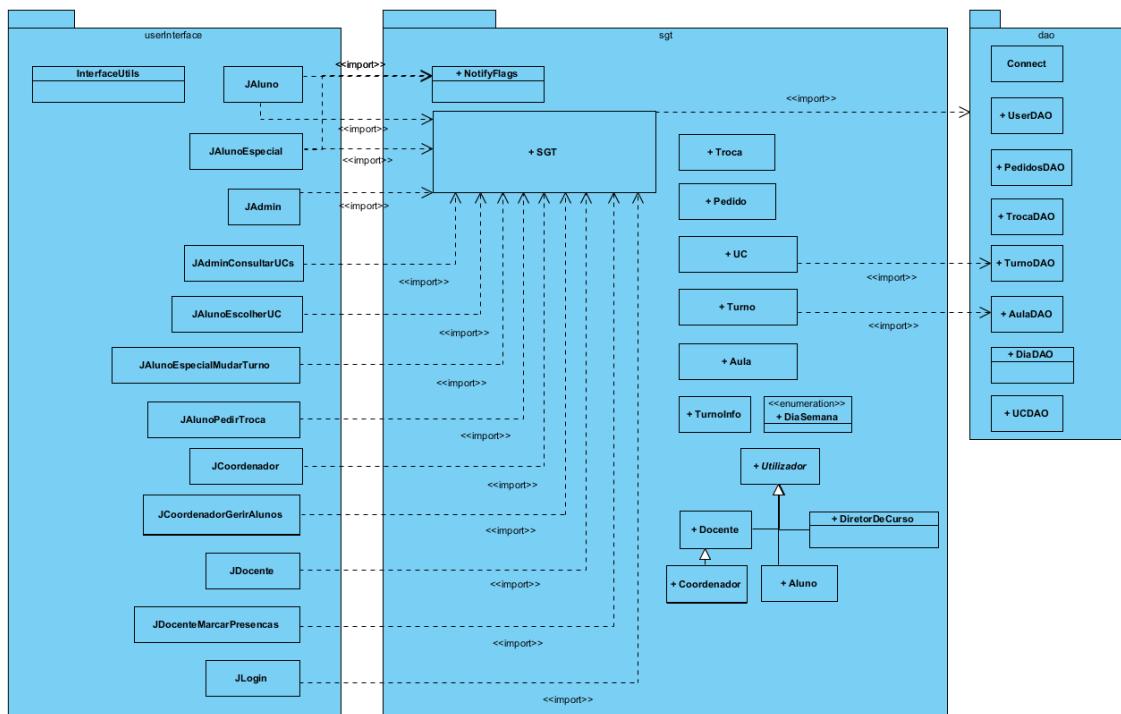


Figura 6.1: diagrama de package

7. Diagramas de Classes

Depois de analisados os diagramas anteriores, foi elaborado o seguinte Diagrama de Classes Lógico, que representa a estrutura das classes e a relação entre elas, definindo os métodos necessários para o funcionamento do sistema.

A partir deste, foi também concebido o Diagrama de Classes Físico, adicionando os *Data Access Object (DAO)*, que já contempla as alterações necessárias para a incorporação da base de dados. Nesta transição, foi decidido que que os pedidos deviam ser imediatamente carregados da base de dados e, como tal, a acompanhar o *DAO* de pedidos, temos também estes representados em memória.

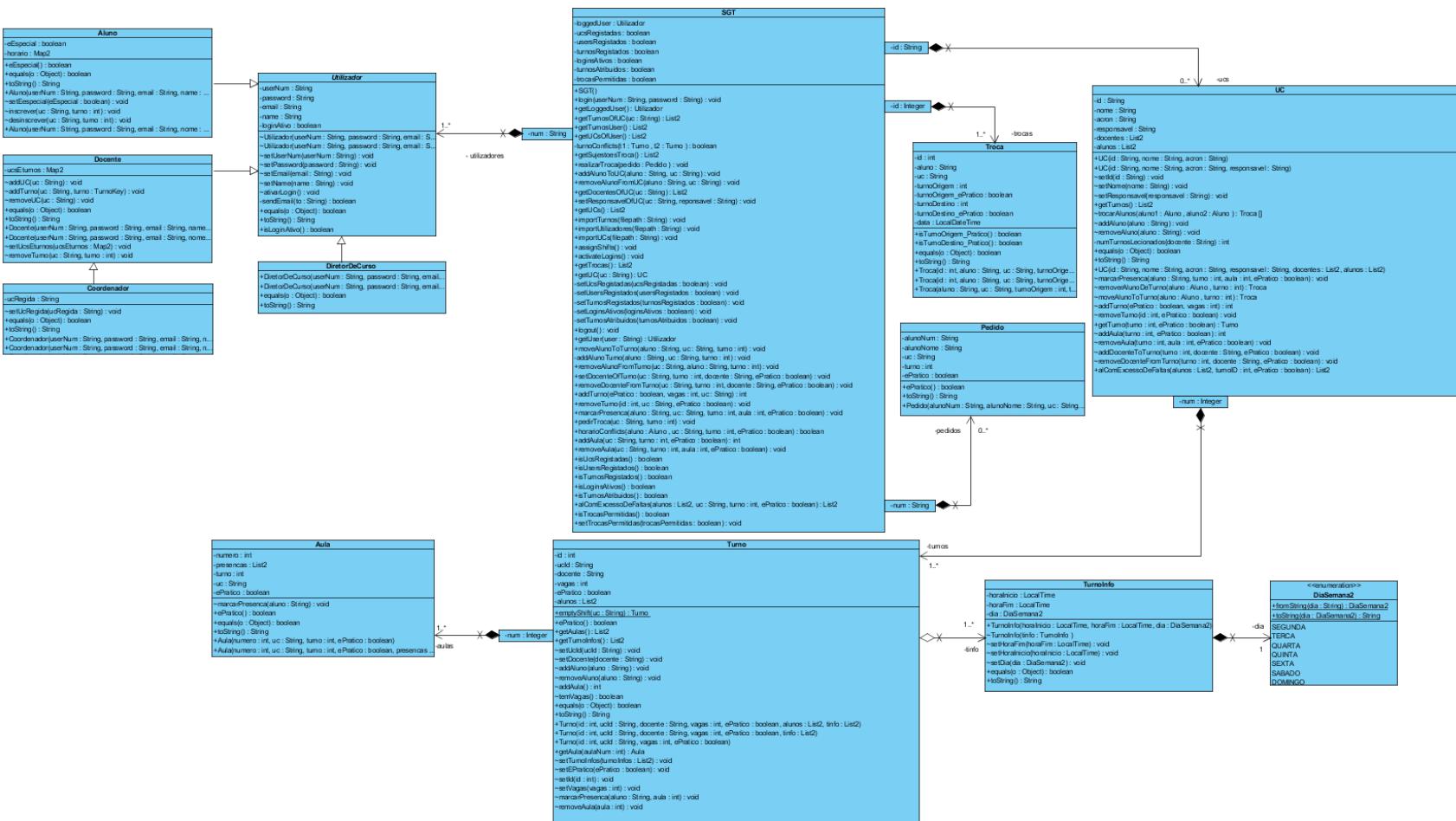


Figura 7.1: Diagrama de Classes Lógico

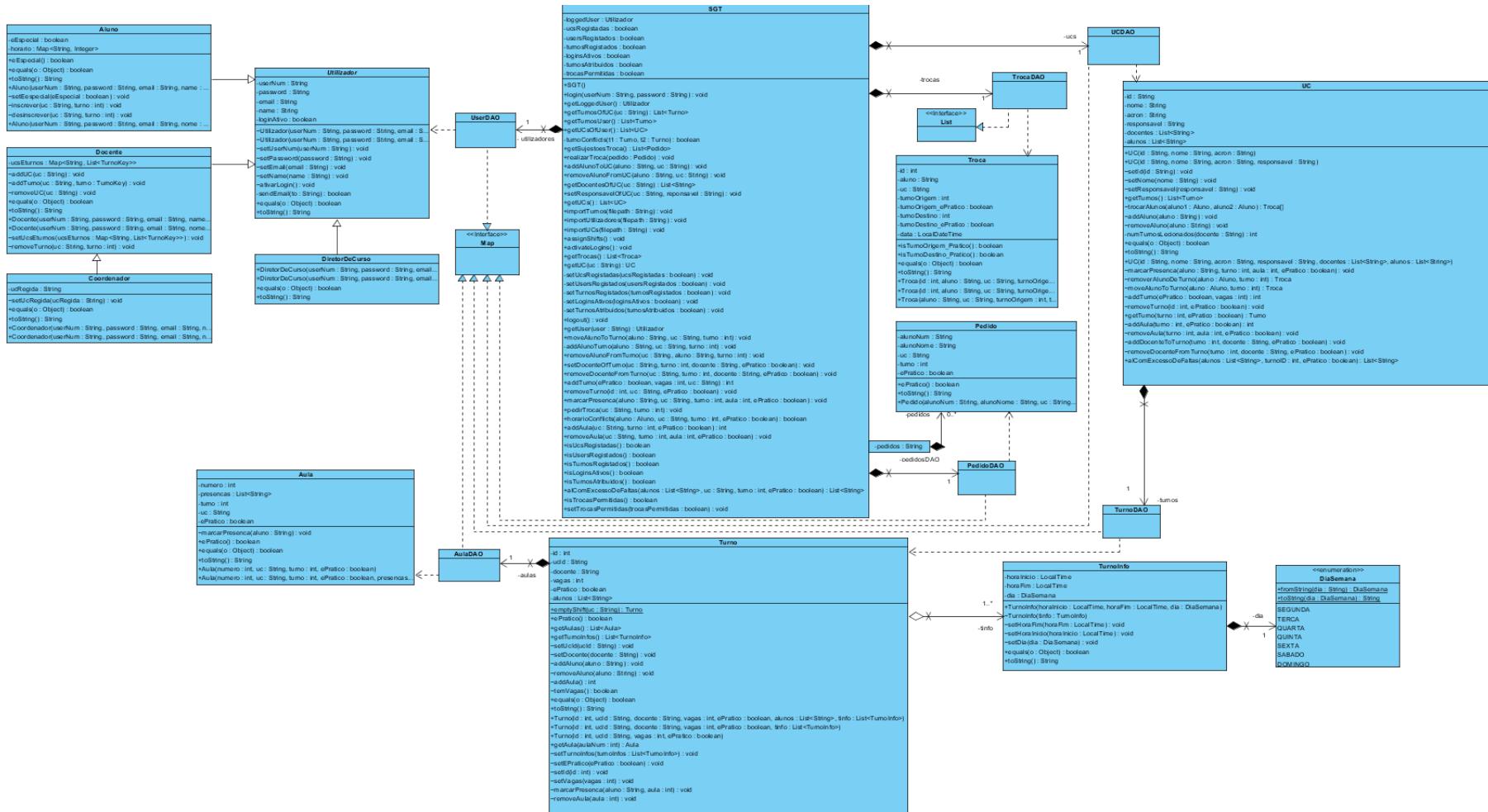


Figura 7.2: Diagrama de Classes

8. Diagramas de Sequência com Subsistemas

No que diz respeito aos Diagramas de Sequência com Submissões, escolhemos fazer apenas os exemplos que seguem em baixo, uma vez que são aqueles que traduzem as funcionalidades mais essenciais da nossa aplicação.

8.1 Realizar Troca

Estes diagramas modelam uma parte essencial da aplicação desenvolvida: a troca de turno entre dois alunos. Esta faz uso de vários outros métodos também modelados para atingir este efeitos. No entanto, alguns deles são usados também noutros *use cases*. Por exemplo, o **addAluno** e o **removeAluno** (fig 8.4 e 8.5) são utilizadas, respectivamente, quando o Coordenador deseja adicionar ou retirar um aluno de um turno.

No que respeita ao **realizarTrocada** em si, primeiro infere-se, através do Pedido, a UC dentro da qual a troca será efetuada e, de seguida, trocam-se os alunos entre si, movendo cada um para o turno do outro.

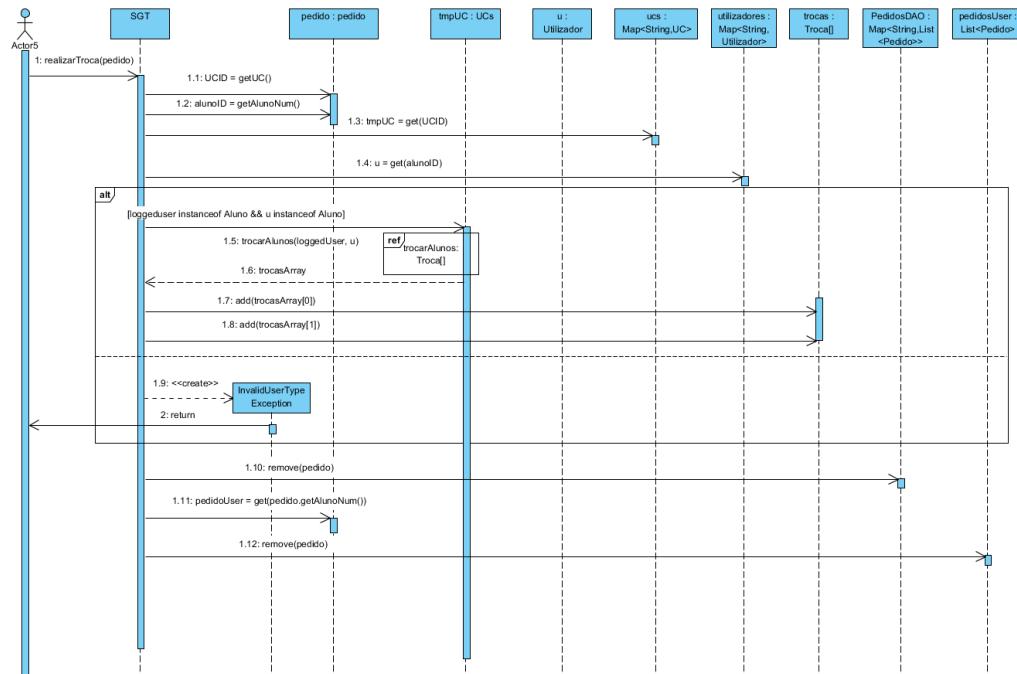


Figura 8.1: Diagrama de sequencia do método SGT#realizarTrocada

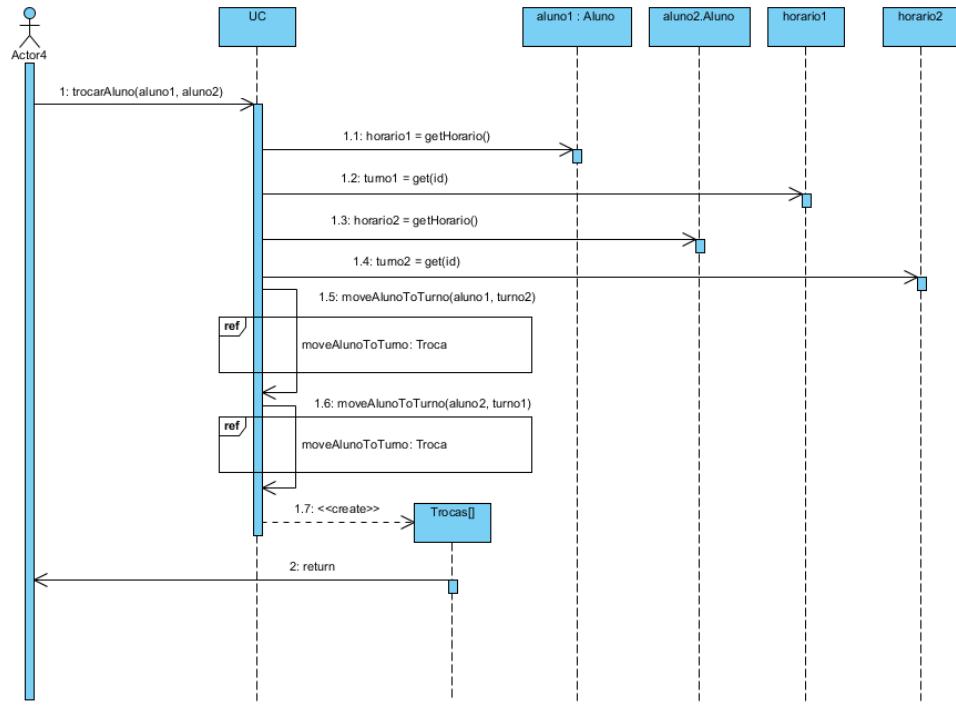


Figura 8.2: Diagrama de sequencia do método UC#trocarAlunos

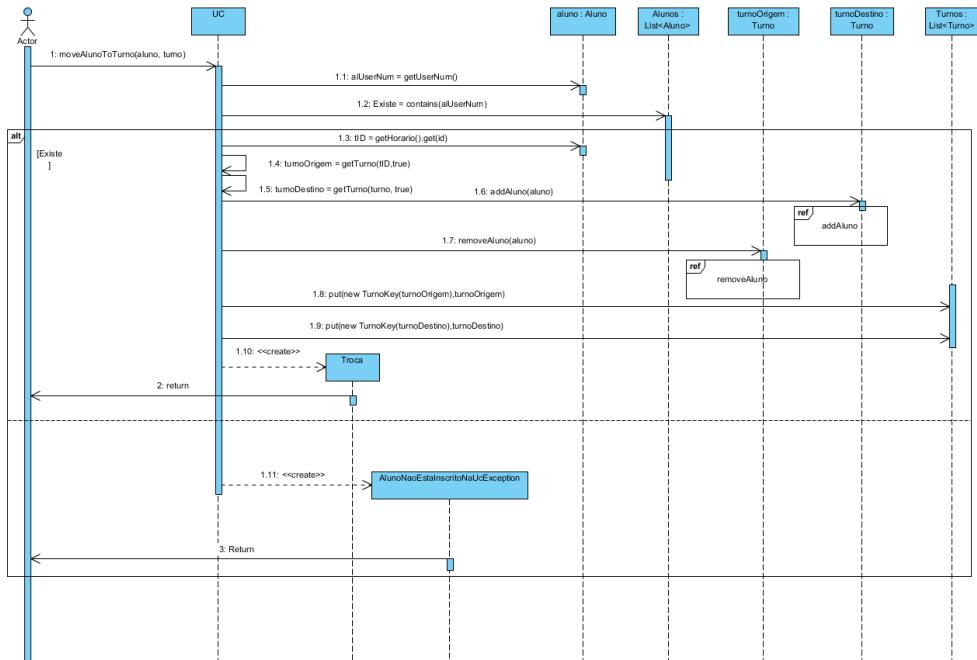


Figura 8.3: Diagrama de sequencia do método UC#moveAlunoToTurno

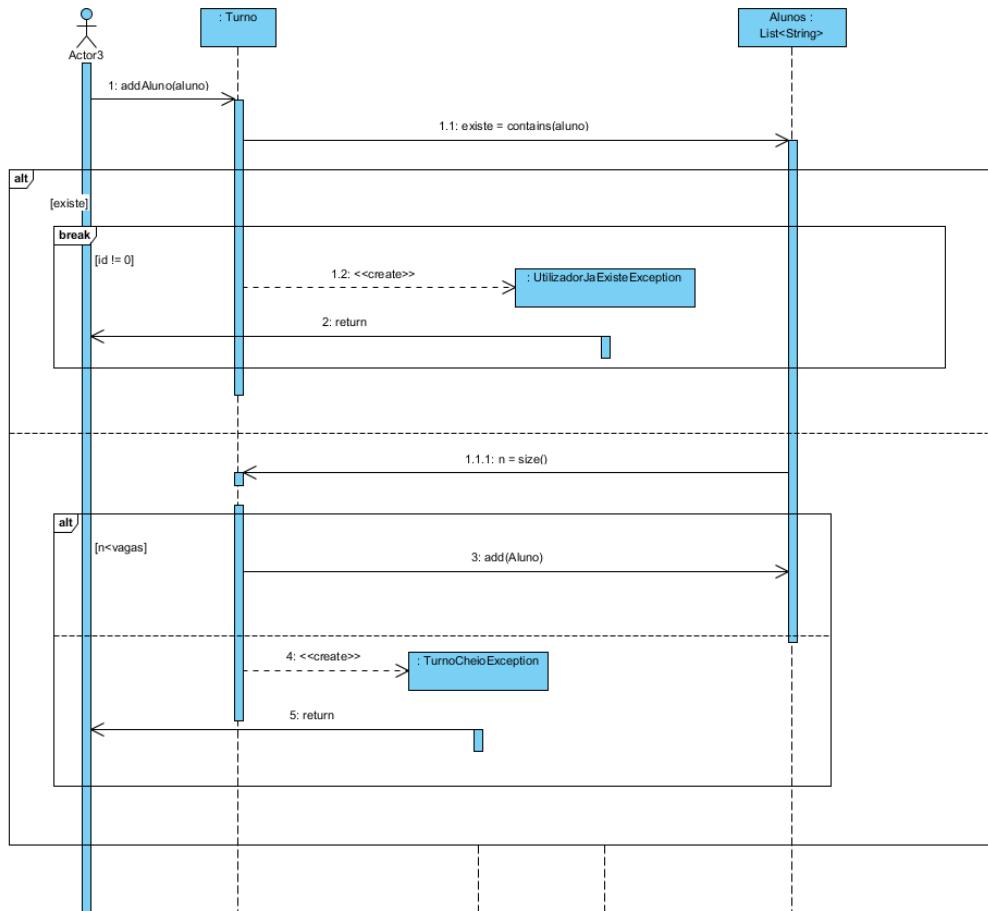


Figura 8.4: Diagrama de sequencia do método Turno#addAluno

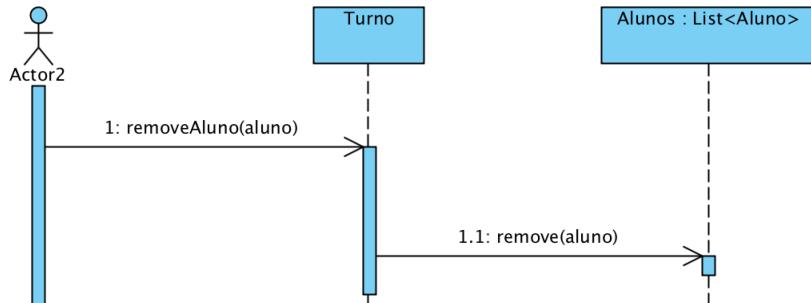


Figura 8.5: Diagrama de sequencia do método Turno#removerAluno

8.2 Atribuir Turnos

Este diagrama traduz a modelação do método que distribui os turnos pelos alunos. O sistema tenta, para cada UC a que cada aluno está inscrito, encontrar um turno cujo horário lhe permita frequentar as aulas.

Estão também modelados alguns métodos auxiliares, como o **horarioConflicts** e a seu auxiliar **turnoConflicts** (8.7 e 8.8), que verificam, respetivamente, se um turno tem sobreposição com um horário (conjunto de turnos) e se um turno tem sobreposição com outro. Este métodos são depois usados pela interface gráfica para

avisar o utilizador que está a tentar mudar para um turno que entra em conflito com o seu horário.

Se não houver vagas em nenhum dos turnos de uma UC, o sistema cria uma exceção para reportar esta situação. Da mesma forma, se, para algum aluno, não existir um turno em que ele possa ser inserido, o método cria uma exceção para comunicar o problema.

Ao ser implementado o método sofreu pequenas alterações. Foi tomada a decisão de ordenar as UCs e alunos, respeitando os seguintes critérios de ordenação: para as UC, primeiro considera-se o decrescente número de aulas que cada turno tem por semana e, depois, o crescente número de turnos; para os alunos, considera-se o número decrescente de UCs a que estão inscritos.

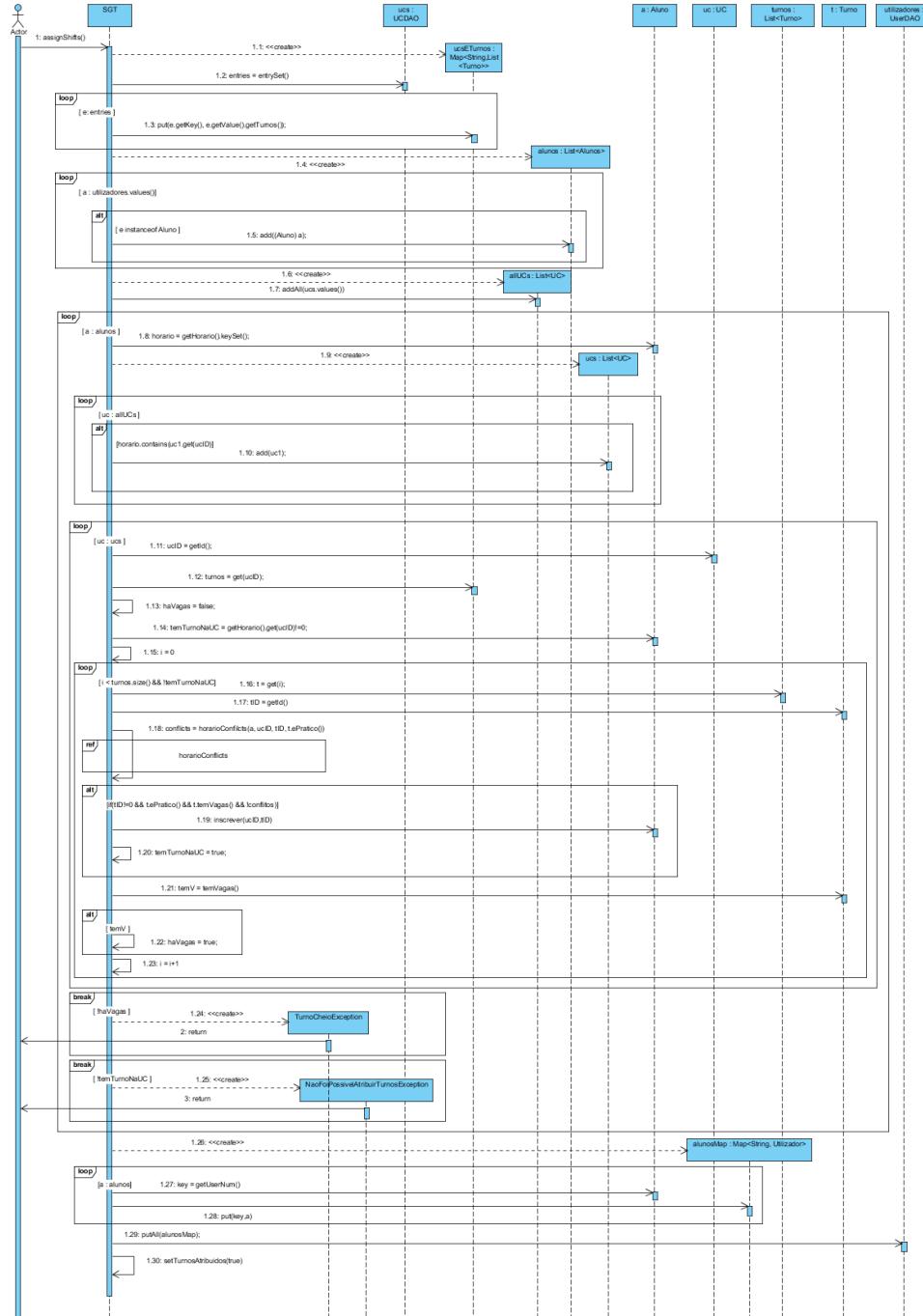


Figura 8.6: Diagrama de sequencia do método `SGT#assignShifts`

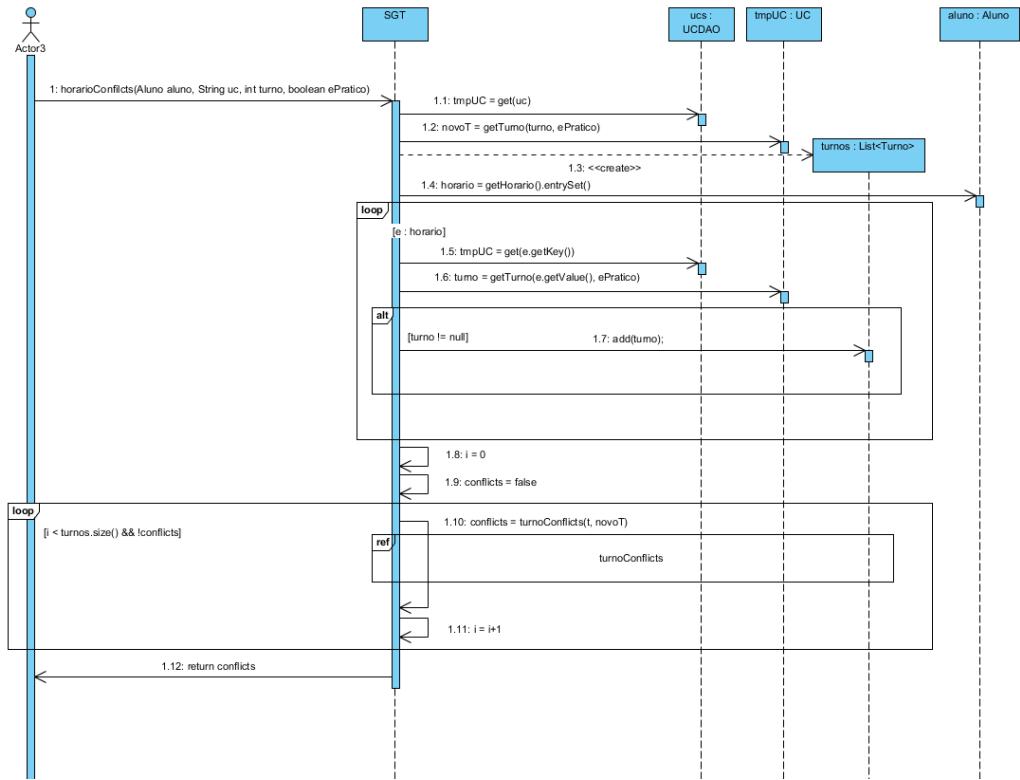


Figura 8.7: Diagrama de sequencia do método SGT#horarioConflicts

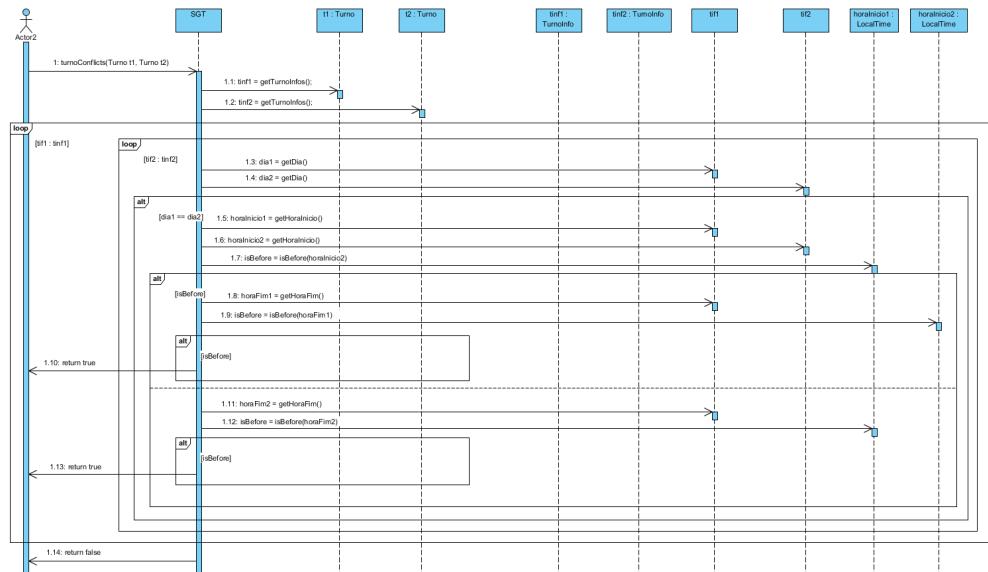


Figura 8.8: Diagrama de sequencia do método SGT#turnoConflicts

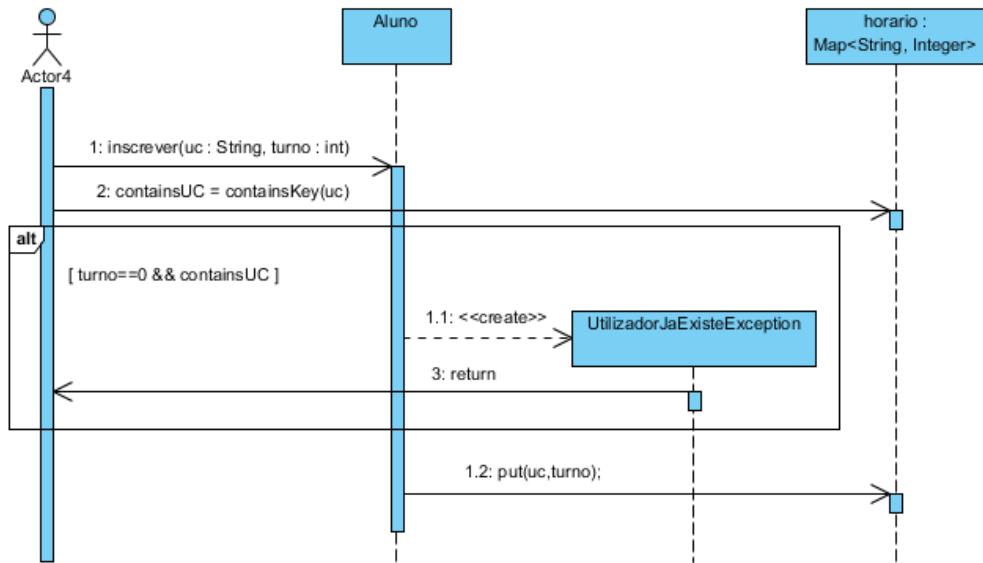


Figura 8.9: Diagrama de sequencia do método `Aluno#inscrever`

9. Diagrama de Instalação

Tendo em conta que a aplicação desenvolvida foi planeada para apenas interagir com a base de dados e que é instalada numa única máquina, ou seja, numa unica máquina (*localhost*) temos apenas dois componentes: a aplicação desenvolvida em Java e a base de dados desenvolvida em MySQL, pelo que o Diagrama de Instalação se traduz no seguinte:

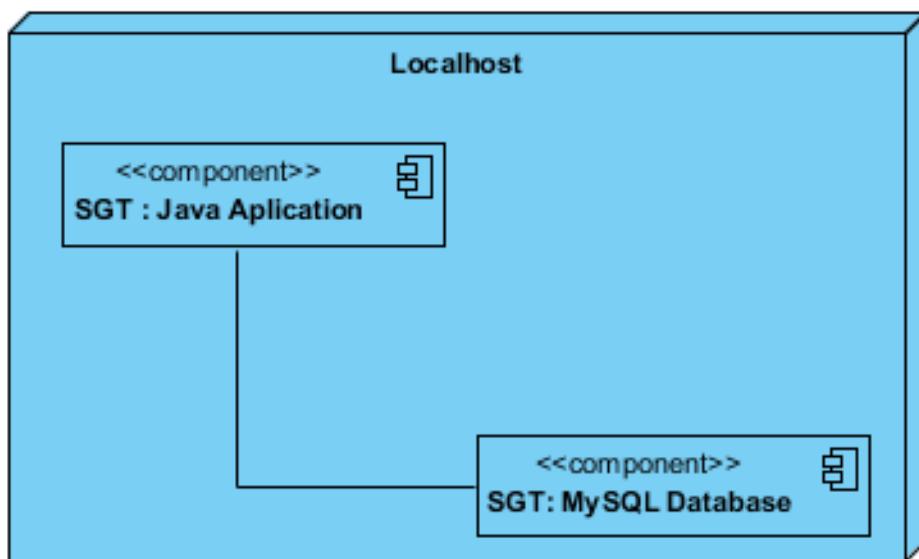


Figura 9.1: Diagrama de Instalação

10. Base de Dados

Para suportar a aplicação, foi desenvolvida a Base de Dados representada pelo seguinte modelo lógico.

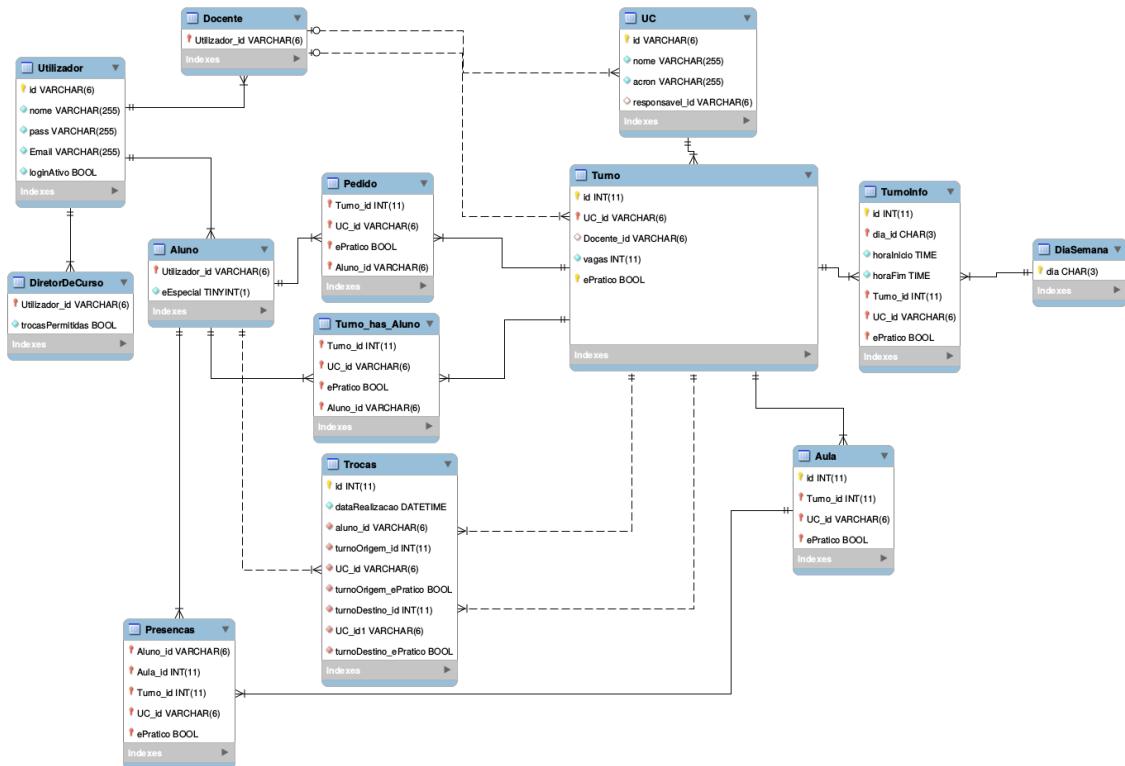


Figura 10.1: Modelo Lógico da Base de Dados

11. Conclusão e trabalho futuro

Com a realização deste trabalho, podemos concluir que a modelação de uma aplicação tem uma importância tão grande quanto a escrita do seu código fonte, permitindo uma simplificação e melhor compreensão por parte de quem a desenvolve. Sem esta, a sua implementação seria muito mais complicada.

Através da realização de um diagrama de domínio somos capazes de traduzir o problema proposto. Da mesma forma, a especificação dos use cases ajuda-nos a perceber quais as funcionalidades da aplicação e as permissões de cada utilizador. Os diagramas de classe servem como um valioso *mapa* do sistema, na medida em que permitem compreender melhor as responsabilidades de cada classe. Os diagramas de sequência de sistema servem como auxiliares para a construção do código, uma vez que neles é possível definir os métodos e a sequência das operações que representam as funcionalidades da aplicação. Por fim o diagrama de instalação, neste caso, traduz a ligação entre a base de dados e a aplicação em Java.

De forma geral, é possível concluir que o objetivo que nos foi proposto foi alcançado com sucesso. Foi desenvolvida uma aplicação que responde aos requisitos impostos e que permite uma gestão eficiente dos turnos. No entanto, há um aspeto que não foi implementado com sucesso: a visualização do histórico dos turnos que o diretor de curso seria capaz de fazer, pois, apesar deste histórico ser mantido pelo sistema, não o integramos na interface gráfica e, portanto, não é possível de visualizar. Poderiam ainda ser melhorados alguns aspetos da interface gráfica, de forma a torná-la mais *user friendly*.

A. Especificação dos Use Cases

A.1 Login

Use Case: Login		
Descrição: Utilizador efetua login		
Pré-condição: --		
Pós-condição: Utilizador fica autenticado.		
	Actor	Sistema
Comportamento Normal	1. Inserir credenciais	
		1. Valida credenciais
		2. Autentica o utilizador
		3. Informa de sucesso
Exceção 1 [Credenciais Inválidas] (passo 1)		1.1. Informa que o email ou password não são válidos

Figura A.1: Especificação do *Use Case* Login

A.2 Escolher UCs

Use Case: Escolher UCs		
Descrição: Um aluno escolhe as UCs em que se quer inscrever		
Pré-condição: O utilizador está autenticado como aluno e não tem turnos atribuídos.		
Pós-condição: O aluno fica inscrito às UCs que selecionou		
	Actor	Sistema
Comportamento Normal	1. Indica uma UC em que se pretende inscrever	
		2. Verifica se existe
		3. Regista a UC pretendida
		4. Informa que a operação foi concluída com sucesso.
Exceção 1 [UC não existe] (passo 2)		2.1. Informa que a UC não existe

A.3 Consultar UCs em que está inscrito

Use Case: Consultar UCs em que está inscrito		
Descrição: Aluno solicita a consulta de UCs a que está inscrito		
Pré-condição: O utilizador está autenticado como Aluno.		
Pós-condição: --		
	Actor	Sistema
Comportamento Normal	1. Solicita visualização de UCs a que está inscrito.	
		2. Cria lista de UCs.
		3. Mostra as UCs a que o aluno está inscrito

A.4 Consultar os turnos que lhe foram atribuídos

Use Case: Consultar os turnos que lhe foram atribuídos		
Descrição: Aluno solicita a consulta dos turnos que lhe foram atribuídos		
Pré-condição: O utilizador está autenticado como Aluno e tem turnos atribuídos.		
Pós-condição: --		
	Actor	Sistema
Comportamento Normal	1. Solicita visualização dos turnos que lhe foram atribuídos.	
		2. Cria lista de turnos
		3. Mostra os turnos que estão atribuídos ao aluno.

A.5 Aceitar sugestão de troca

Use Case: Aceitar sugestão de troca		
Descrição: Utilizador aceita uma sugestão de troca do sistema		
Pré-condição: O Utilizador está autenticado como Aluno, tem turnos atribuídos, tem pelo menos uma sugestão de troca e o período de aulas ainda não começou.		
Pós-condição: Dois alunos trocam de turno.		
	Actor	Sistema
Comportamento Normal	1. Indica a sugestão que pretende aceitar.	
		2. Verifica que a troca ainda é possível
		3. Regista a troca.
		4. Informa de sucesso
Exceção 1 [A troca já não é possível] (passo 2)		2.1. Informa que a troca já não é possível.

A.6 Mudar de turno

Use Case: Mudar de turno		
Descrição: Utilizador muda para um turno com vagas		
Pré-condição: O Utilizador está autenticado como Aluno com estatuto especial e tem turnos atribuídos.		
Pós-condição: O Utilizador mudou de turno.		
	Actor	Sistema
Comportamento Normal	1. Indica o turno para onde pretende ir.	
		2. Verifica que o turno tem vagas.
		3. Regista a mudança.
		4. Informa de sucesso
Exceção 1 [A troca já não é possível] (passo 2)		2.1. Informa que o turno está cheio.

A.7 Marcar presenças

Use Case: Marcar Presenças		
Descrição: Utilizador marca presenças		
Pré-condição: O Utilizador está autenticado como Docente e as aulas já começaram.		
Pós-condição: As presenças de uma aula ficam atualizadas.		
	Actor	Sistema
Comportamento Normal	1. Indica o turno e a aula que quer atualizar.	
	2. Indica os alunos que estiveram presentes na aula.	
	3. Indica os alunos externos ao turno que estiveram presentes	
		4. Regista as presenças.

A.8 Consultar turnos que leciona

Use Case: Consulta turnos que leciona		
Descrição: O Docente consulta os turnos que leciona.		
Pré-condição: O Utilizador está autenticado como Docente e as aulas já começaram.		
Pós-condição: --		
	Actor	Sistema
Comportamento Normal	1. Solicita a visualização dos turnos que leciona.	
		2. Apresenta os turnos que o Docente leciona e a composição dos mesmos.

A.9 Adicionar aluno a um turno

Use Case: Adicionar aluno a um turno.		
Descrição: O Docente adiciona um aluno a um turno.		
Pré-condição: O Utilizador está autenticado como Coordenador e o período de aulas já começou.		
Pós-condição: Um aluno foi adicionado a um turno.		
	Actor	Sistema
Comportamento Normal	1. Indica o turno para onde pretende adicionar o aluno.	
		2. Verifica que o turno tem vagas.
	3. Indica o aluno que pretende adicionar.	
		4. Adiciona o aluno ao turno.
		5. Informa de sucesso.
Exceção 1 [O turno está cheio] (passo 2)		2.1. Informa que o turno está cheio.
Exceção 2 [O aluno já está no turno] (passo 3)		2.2. Informa que o aluno já está no turno.

A.10 Remover aluno a um turno

Use Case: Remover aluno de um turno		
Descrição: Docente remove um aluno de um turno.		
Pré-condição: O Utilizador está autenticado como Coordenador e o período de aulas já começou.		
Pós-condição: Um aluno é retirado de um turno.		
	Actor	Sistema
Comportamento Normal	1. Indica o aluno que pretende remover.	
		2. Remove o aluno.
		3. Informa de sucesso
Exceção 1 [O aluno não está no turno] (passo 2)		2.1. Informa que o aluno não pertence ao turno.

A.11 Importar UCs

Use Case: Importar UCs		
Descrição: O Diretor de curso importa as UCs de um ficheiro.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e os alunos ainda não podem fazer log in.		
Pós-condição: As UCs são importadas		
	Actor	Sistema
Comportamento Normal	1. Indica o ficheiro de onde quer importar.	
		2. Verifica que a sintaxe do ficheiro está correta.
		3. Importa e regista os dados.
		4. Informa que a operação foi bem-sucedida.
Exceção 1 [Sintaxe errada] (passo 2)		2.1 Informa que o ficheiro tem a sintaxe errada.
Alternativa 1 [O sistema já tem UCs importadas] (passo 3)		3.1 Informa que vai fazer <i>overwrite</i>
	3.2 Indica que quer prosseguir mesmo assim.	
		3.3. Regressa ao passo 3.
Exceção 2 [O DC não quer fazer <i>overwrite</i> às UCs] (passo 3.2)		3.2.1 Informa que a operação foi cancelada.

A.12 Importar alunos

Use Case: Importar alunos		
Descrição: O Diretor de curso importa os alunos de um ficheiro.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e os alunos ainda não podem fazer log in.		
Pós-condição: Os alunos são importados		
	Actor	Sistema
Comportamento Normal	1. Indica o ficheiro de onde quer importar.	
		2. Verifica que a sintaxe do ficheiro está correta.
		3. Importa e regista os dados.
		4. Informa que a operação foi bem-sucedida.
Exceção 1 [Sintaxe errada] (passo 2)		2.1 Informa que o ficheiro tem a sintaxe errada.
Alternativa 1 [O sistema já tem alunos importados] (passo 3)		3.1 Informa que vai fazer <i>overwrite</i>
	3.2 Indica que quer prosseguir mesmo assim.	
		3.3. Regressa ao passo 3.
Exceção 2 [O DC não quer fazer <i>overwrite</i> aos alunos] (passo 3.2)		3.2.1 Informa que a operação foi cancelada.

A.13 Importar turnos

Use Case: Importar turnos		
Descrição: O Diretor de curso importa os turnos de um ficheiro.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e os alunos ainda não tem turnos atribuídos.		
Pós-condição: Os turnos são importados		
	Actor	Sistema
Comportamento Normal	1. Indica o ficheiro de onde quer importar.	
		2. Verifica que a sintaxe do ficheiro está correta.
		3. Importa e regista os dados.
		4. Informa que a operação foi bem-sucedida.
Exceção 1 [Sintaxe errada] (passo 2)		2.1 Informa que o ficheiro tem a sintaxe errada.
Alternativa 1 [O sistema já tem turnos importados] (passo 3)		3.1 Informa que vai fazer <i>overwrite</i>
	3.2 Indica que quer prosseguir mesmo assim.	
		3.3. Regressa ao passo 3.
Exceção 2 [O DC não quer fazer <i>overwrite</i> aos turnos] (passo 3.2)		3.2.1 Informa que a operação foi cancelada.

A.14 Ativar logins

Use Case: Ativar Logins		
Descrição: O Diretor de curso ativa os logins enviando para todos os alunos uma password gerada pelo sistema.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e há UCs e alunos registados.		
Pós-condição: Os alunos podem efetuar log in.		
	Actor	Sistema
Comportamento Normal	1. Indica que pretende ativar os logins.	
		2. Verifica que todos os alunos têm um endereço de email associado.
		3. Para cada aluno gera uma password e envia-a por email.
		4. Informa que a operação foi concluída com sucesso.
Exceção 1 [Pelo menos um aluno não tem email associado] (passo 2)		3.1 Informa que a operação foi cancelada porque pelo menos um aluno não tem email associado.

A.15 Apagar turno

Use Case: Apagar turno		
Descrição: O Diretor de curso apaga um turno.		
Pré-condição: O Utilizador está autenticado como Diretor de curso.		
Pós-condição: O turno é apagado		
	Actor	Sistema
Comportamento Normal	1. Indica o turno que quer apagar.	
		2. Verifica que o turno está vazio.
		3. Apaga o turno.
Exceção 1 [O turno não existe] (passo 1)		2.1 Informa que o turno não existe.
Exceção 2 [O turno não está vazio] (passo 2)		2.1 Informa que o turno não pode ser apagado por ainda ter alunos.

A.16 Adicionar turno

Use Case: Adicionar Turno		
Descrição: O Diretor de curso adiciona um turno.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e há UCs registadas.		
Pós-condição: O turno é adicionado.		
	Actor	Sistema
Comportamento Normal	1. Indica a UC do turno.	
		2. Verifica que a UC existe.
	3. Indica as horas do turno	
		4. Regista o novo turno.
		5. Informa que a operação foi concluída com sucesso.
Exceção 1 [A UC não existe] (passo 2)		2.1 Informa que a UC não existe.

A.17 Consultar UC

Use Case: Consultar UC		
Descrição: O Diretor de curso consulta uma UC.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e há UCs registadas.		
Pós-condição: --		
	Actor	Sistema
Comportamento Normal	1. Indica a UC quer consultar	
		2. Verifica que a UC existe.
		3. Apresenta a UC
Exceção 1 [A UC não existe] (passo 2)		2.1 Informa que a UC não existe.

A.18 Consultar turnos

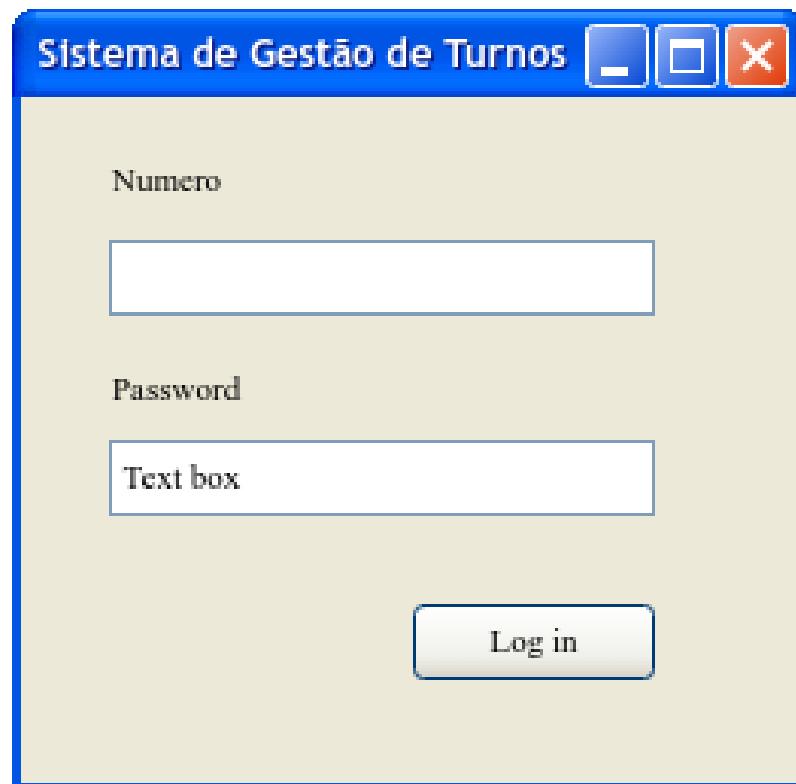
Use Case: Consultar turno		
Descrição: O Diretor de curso consulta um turno.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e há UCs registadas.		
Pós-condição: --		
	Actor	Sistema
Comportamento Normal		1. <<include>> Consultar UCs
	2. Indica o turno que quer consultar	
		3. Verifica que o turno existe
		4. Apresenta o turno
Exceção 1 [O turno não existe] (passo 2)		2.1 Informa que o turno não existe.

A.19 Atribuir Turnos

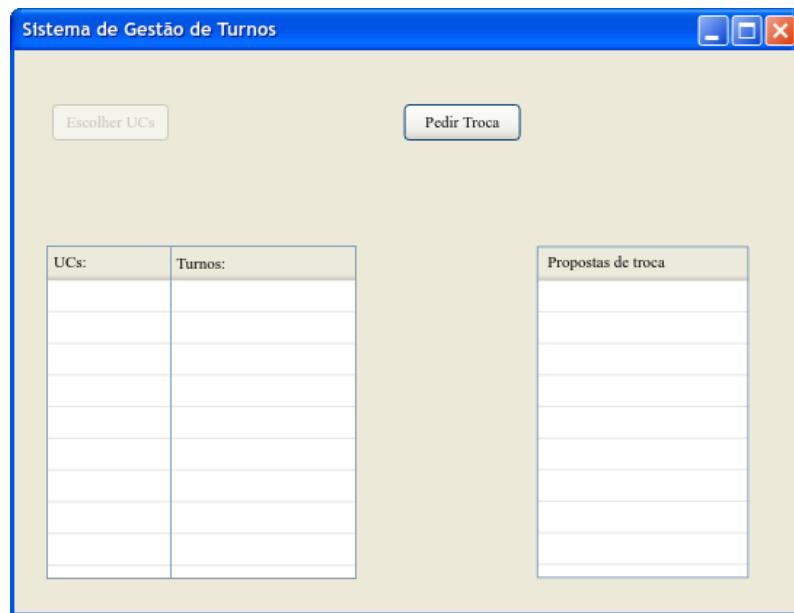
Use Case: Atribuir turnos		
Descrição: O Diretor de curso atribui turnos aos alunos.		
Pré-condição: O Utilizador está autenticado como Diretor de curso e há UCs, turnos e alunos registados.		
Pós-condição: Os alunos ficam com turnos atribuídos.		
	Actor	Sistema
Comportamento Normal	1. Indica que pretende atribuir os turnos aos alunos.	
		2. Pede confirmação ao utilizador
	3. Indica que pretende prosseguir	
		4. Atribui os turnos automaticamente
		5. Informa que a operação foi concluída com sucesso.
Exceção 1 [O utilizador não quer prosseguir] (passo 3)		3.1 Informa que a operação foi cancelada.
Exceção 2 [Os turnos não tem capacidade para os alunos todos] (passo 4)		4.1 Informa que a operação foi cancelada porque não há capacidade para todos os alunos.
Exceção 3 [Não é possível atribuir todos os turnos sem haver conflitos] (passo 4)		4.1 Informa que a operação foi cancelada por não conseguir que os turnos fossem atribuídos sem haver conflitos.

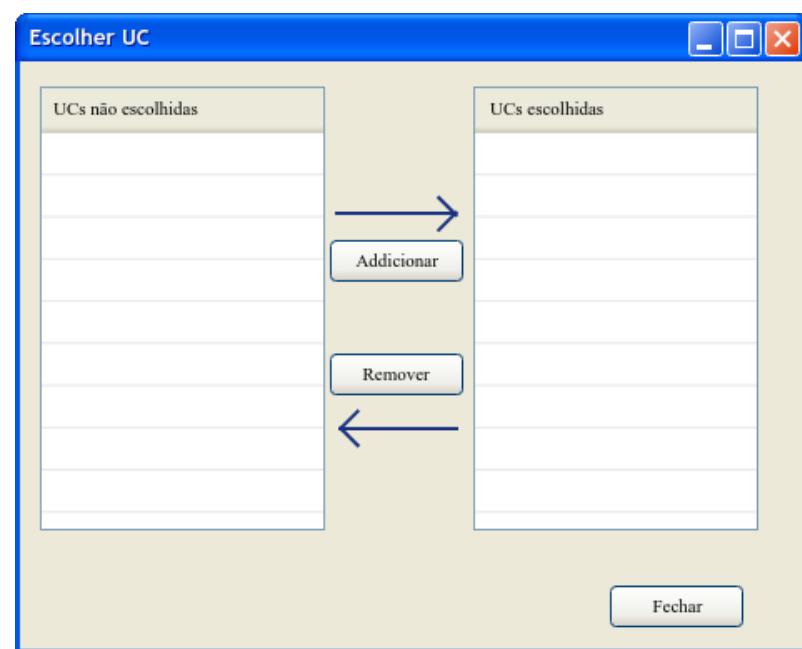
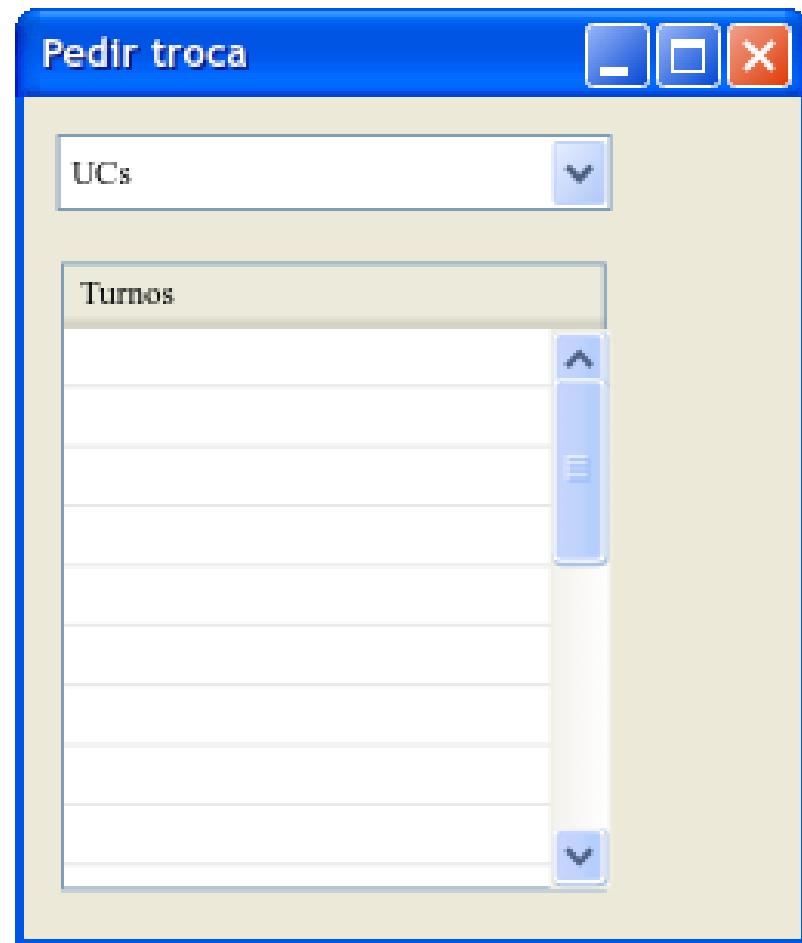
B. *Mockups*

B.1 Utilizador

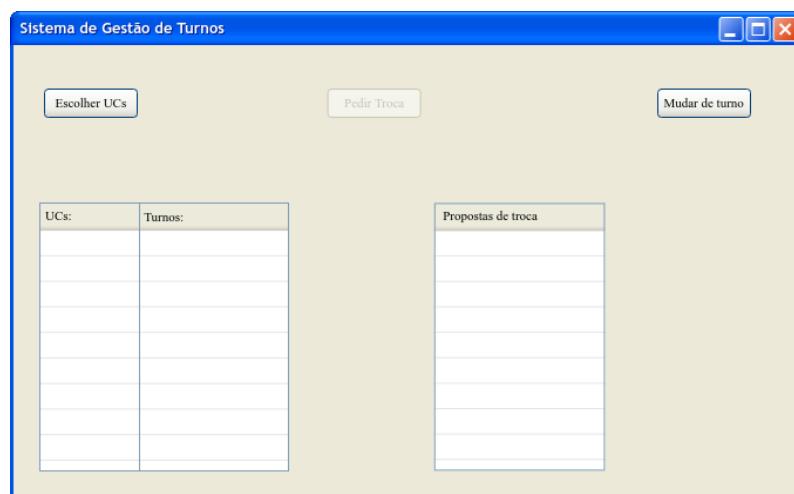
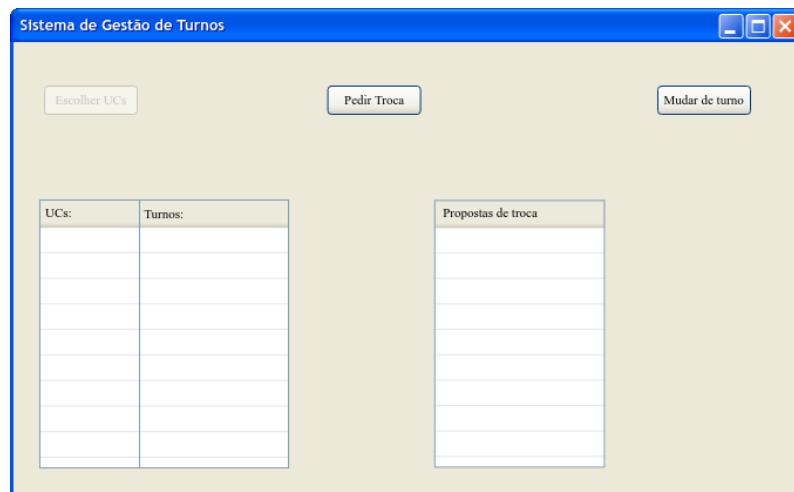


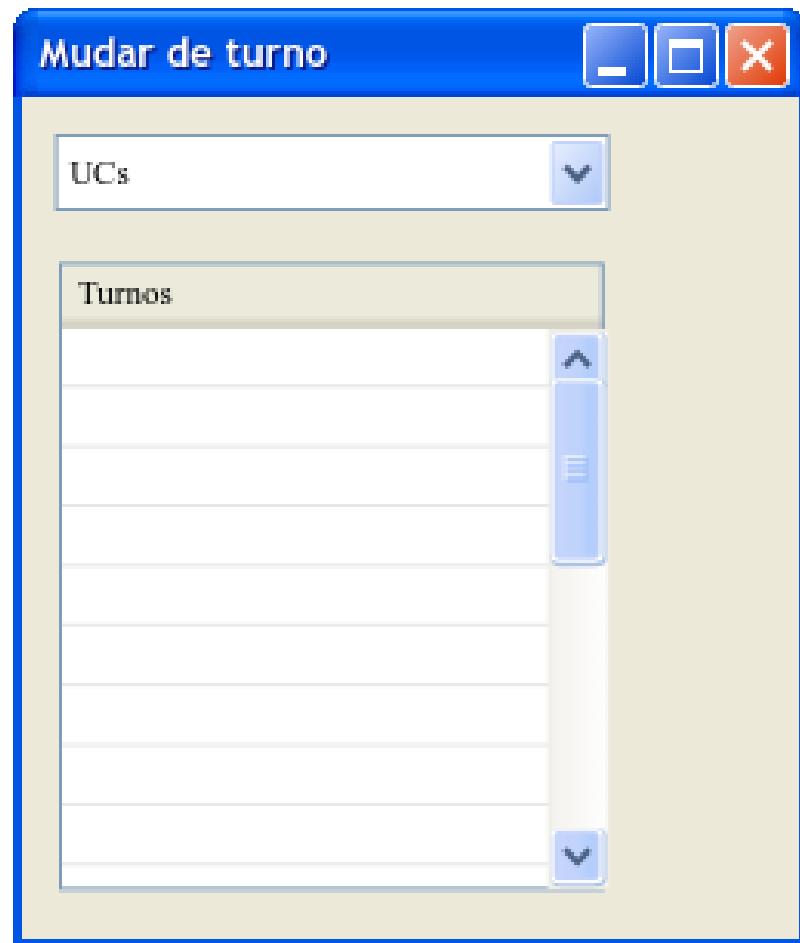
B.2 Aluno



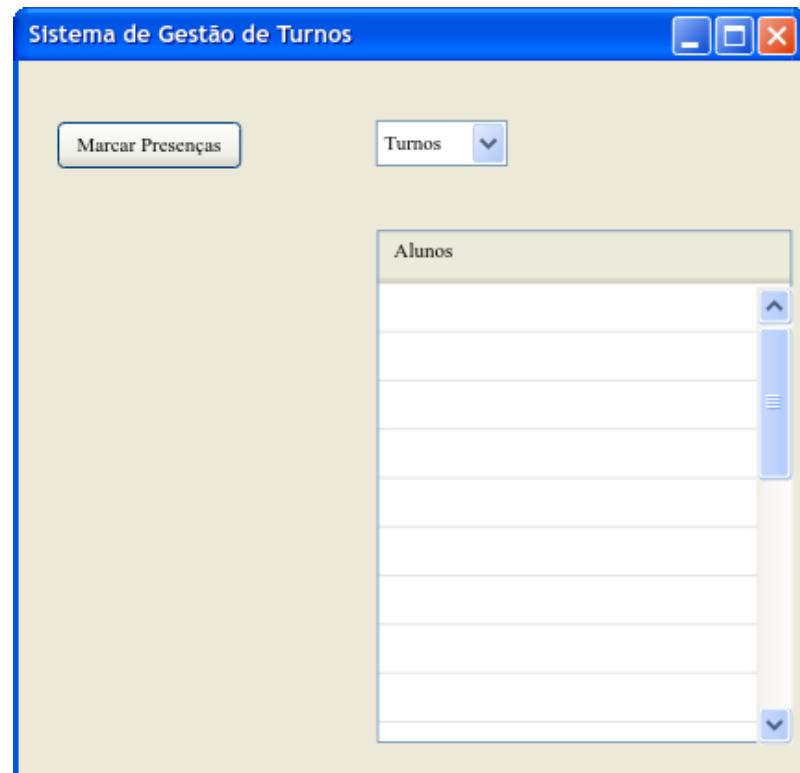


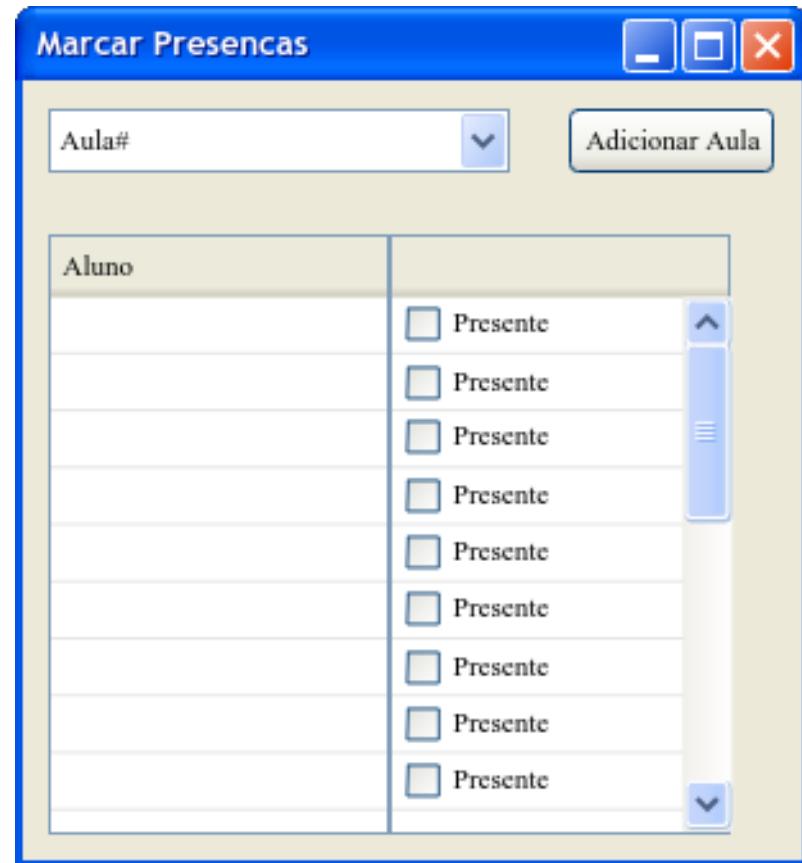
B.3 Aluno com estatuto especial





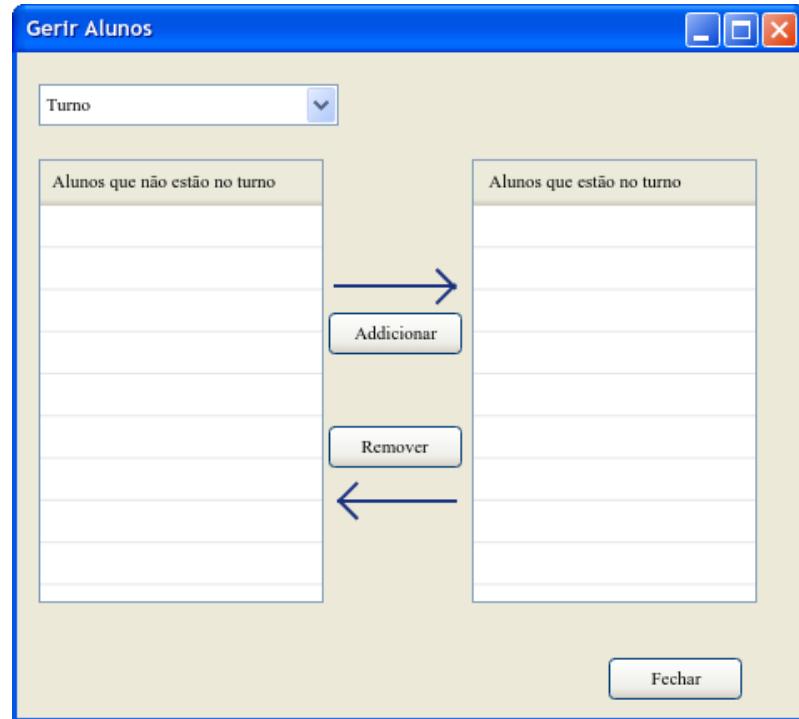
B.4 Docente





B.5 Coordenador





B.6 Diretor de Curso





