**Universidade do Minho**

**Departamento de Informática**

**Desenvolvimento de Sistemas de Software**

**Relatório de Projeto – Fase 1**



Carlos Pereira – A61887



João Pires Barreira – A73831



João Gomes – A74033



João Reis – A75372

**MIEI - 2016/2017**

**Introdução**

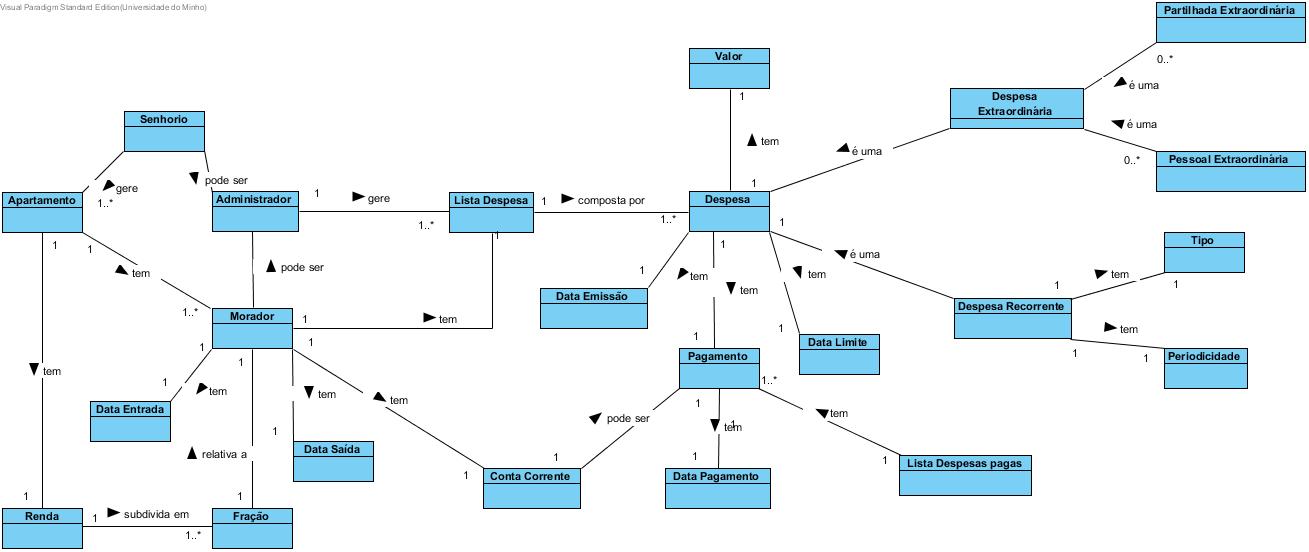
Este trabalho visa o desenvolvimento de um sistema de suporte à partilha de despesas num apartamento. Tem como objetivo desenvolver uma aplicação que deverá suportar o registo das despesas e a gestão do seu pagamento por parte dos moradores. A partilha de despesas num apartamento faz parte do dia-a-dia dos estudantes universitários, sendo um problema que por vezes é difícil de lidar, tornando importante o desenvolvimento desta aplicação, que fornece um melhor controlo sobre as despesas de cada morador.

Sabemos que temos dois tipos de despesa: recorrente e extraordinária. Uma despesa é recorrente quando tem uma periodicidade associada a ela. Ao fim de um período de tempo tem que ser acrescentada e paga (por exemplo, renda, conta da luz, conta da água, etc.). Fora estas despesas, todas as outras são extraordinárias. Ou seja, surgem porque existe uma razão para tal (reparar a porta do apartamento ou arranjar uma janela). Também sabemos que as despesas terão de ser divididas pelos moradores. É a partir destas informações que vamos tentar enquadrar o nosso programa num contexto real.

Neste relatório vai ser apresentado o diagrama de Modelo de Domínio, bem como o diagrama de Use Cases e a identificação/descrição dos Use Cases mais relevantes no desenvolvimento da aplicação. Além disso vão ser apresentadas algumas das nossas decisões ao realizar estes diagramas.

**Modelo de Domínio**

O Modelo de Domínio inclui todas as entidades que pensamos serem necessárias para o nosso projeto, nomeadamente: Senhorio, Morador, Apartamento, Despesa, etc. Podemos dizer que as nossas entidades estão divididas em três grupos relacionados entre si: administradores, despesas e pagamentos. Optámos por dividir as entidades de maneira a conseguirmos planear melhor a forma como pretendemos que o nosso programa funcione.



**Figura 1** - Modelo de Domínio

* **Administrador**

As entidades que fazem parte desta categoria são: Apartamento, Senhorio, Administrador, Morador, Data Entrada, Data Saída, Renda e Fração.

Tal como nos é pedido, pretendemos que o nosso programa divida as despesas de um apartamento pelos moradores que lá estão. Todas as despesas são inseridas na Lista de Despesas pelo Administrador que poderá ser o Senhorio ou um Morador do apartamento. Contudo, apenas o Senhorio poderá adicionar (ou remover) um Morador de um Apartamento. Também é o Senhorio que define o valor da Renda total de um Apartamento.

Achamos importante que o nosso programa seja flexível ao ponto de permitir que os moradores paguem valores de renda diferentes uns dos outros. Ou seja, por exemplo, faz sentido que no caso de haver um Morador com casa-de-banho privativa tenha um valor de renda superior em comparação com os restantes. Por isso é que incluímos no Modelo de Domínio entidades como Apartamento, Renda e Fração, relacionando-as uma com a outra. Fração é a percentagem da renda total que um morador tem que pagar ao senhorio.

Sempre que há a entrada de um morador num apartamento, o senhorio tem que indicar a data de entrada. Isto é importante para que um morador não seja obrigado a pagar despesas que são anteriores à sua entrada. Quando isto acontece, a percentagem da renda total de todos os moradores é atualizada. O senhorio também tem que indicar a data de saída do morador. Um Morador tem uma Conta Corrente. Isto é, tem um saldo de onde os valores das despesas serão descontados. O morador fica responsável por manter esta conta positiva. Caso o valor de uma despesa seja superior ao valor da conta, esta passa a ter um valor negativo.

* **Despesas**

Dentro desta categoria temos: Lista Despesas, Despesa, Valor, Data Emissão, Data Limite, Despesa Extraordinária, Partilhada Extraordinária, Pessoal Extraordinária, Despesa Recorrente, Tipo e Periodicidade.

A Lista de Despesas é composta por todas as despesas do apartamento. Por sua vez, cada Despesa tem um Valor, uma Data de Emissão, uma Data Limite (de pagamento) e um Pagamento correspondente. As despesas podem ser de dois tipos: Despesa Recorrente e Despesa Extraordinária. Uma Despesa Recorrente é uma despesa que se sabe, à partida, ser necessária pagar ao fim de um período de tempo (Periodicidade). As contas da luz e da água são despesas recorrentes, por exemplo. Uma Despesa Extraordinária são as despesas que não são recorrentes. Ou seja, não se sabe quando poderão surgir. Uma despesa destas pode ser partilhada por todos os moradores do apartamento ou por apenas parte dos moradores.

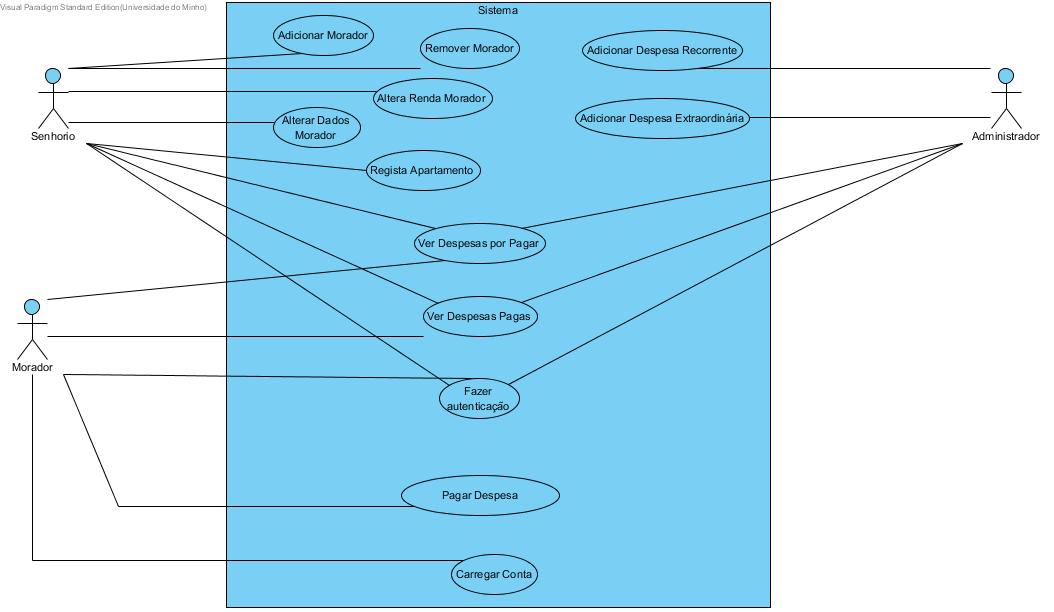
Inicialmente, tínhamos pensado em ter as despesas recorrentes como partilhadas por todos os moradores ou por parte (as mesmas entidades que temos nas despesas extraordinárias), contudo ponderamos melhor e chegámos à conclusão de que não fazia sentido. Isto porque as despesas recorrentes são gerais a todos os moradores, logo iriam ser sempre partilhadas por todos. Sendo assim, preferimos adicionar a periodicidade a estas despesas. Desta forma, o programa vai saber quando deverá acrescentar a despesa aos moradores.

* **Pagamentos**

Aqui temos as entidades: Pagamento, Conta Corrente, Data Pagamento e Lista Despesas Pagas.

Quanto ao pagamento de uma Despesa, este é feito a partir da Conta Corrente de um Morador numa Data de Pagamento. Tal como já foi dito, o Morador fica responsável por gerir o saldo da conta, fazendo os depósitos necessários de forma que haja sempre dinheiro suficiente para pagar as despesas. Quando uma despesa é paga com sucesso, esta é acrescentada à Lista de Despesas Pagas. Desta maneira é possível verificar quais as despesas que um dado morador já pagou.

**Descrição dos Use Case mais relevantes**



**Figura 2** - Modelo de Use Case

1. **Adicionar morador**

Para adicionar um morador ao sistema é necessário que o utilizador autenticado seja o senhorio.

Em primeiro lugar, o ator tem que fornecer os dados do morador ao sistema e que o sistema os valide e registe. Além disso, o sistema incrementa o número de moradores do apartamento, criando e devolvendo os dados de autenticação do novo morador no sistema.

Depois, o ator fornece a fração da renda paga pelo novo morador, sendo que o sistema se encarrega de, automaticamente, registar essa mesma fração, bem como atualizar as frações dos restantes moradores e informar o ator.

Ao serem inseridos os dados do ator no sistema por parte do senhorio, pode ocorrer o caso do morador já exista no apartamento, pelo que o sistema indica o sucedido e o ator regressa ao primeiro passo (podendo indicar novos dados ao sistema).

Além disso, neste mesmo passo, pode ocorrer o caso em que o ator introduza dados inválidos. Caso esse em que o sistema informa o ator e fá-lo regressar ao primeiro passo.

1. **Pagar despesa**

Permite a um morador pagar uma despesa desde que esteja autenticado e tenha despesas por pagar.

Em primeiro lugar, o morador indica os dados da despesa ao sistema que se encarrega de os validar e de apresentar o valor da mesma.

Depois, o ator escolhe pagar a despesa, pelo que o sistema de encarrega de verificar o saldo da usa conta corrente, validar o pagamento, e decrementar o valor da despesa do saldo da conta do morador. Além disso, o sistema remove a despesa da lista de despesas por pagar do morador e adiciona-a à lista de despesas pagas. Por fim, informa o ator de que a sua despesa foi paga com sucesso.

No entanto, quando o ator tenta pagar a sua despesa, pode ocorrer o caso em que não possui saldo suficiente na sua conta corrente. Nesta situação, o sistema limita-se a informar o ator do ocorrido.

1. **Adicionar despesa recorrente**

Este use case refere-se à possibilidade de ser adicionada uma despesa recorrente por parte de um administrador. Para isso, o administrador – que tanto pode ser o senhorio como também um morador que tenha permissões de administração do sistema –, apenas tem de estar autenticado no sistema.

Em primeiro lugar, o ator fornece o valor, o nome, o tipo e a periodicidade da despesa, para além de fornecer a lista dos moradores que devem estar encarregues de pagar a mesma. De seguida, o sistema valida a lista dos moradores e regista a despesa para cada morador, indicando que a despesa foi registada com sucesso.

No entanto, pode ocorrer o facto de que algum dos moradores indicados pelo ator não existam no sistema. Neste caso, o morador apenas se encarrega de informar o ator e de o fazer regressar ao passo anterior onde terá oportunidade de voltar a inserir a lista de moradores abrangidos.

1. **Adicionar despesa extraordinária**

Este use case refere-se à possibilidade de ser adicionada uma despesa extraordinária por parte de um administrador. Para isso, o administrador – que tanto pode ser o senhorio como também um morador que tenha permissões de administração do sistema –, apenas tem de estar autenticado no sistema.

Em primeiro lugar, o ator fornece o valor e o nome da despesa, bem como a lista dos moradores que devem estar encarregues de pagar a mesma. De seguida, o sistema valida a lista dos moradores e regista a despesa para cada morador, indicando que a despesa foi registada com sucesso.

No entanto, pode ocorrer o facto de que algum dos moradores indicados pelo ator não existam no sistema. Neste caso, o morador apenas se encarrega de informar o ator e de o fazer regressar ao passo anterior onde terá oportunidade de voltar a inserir a lista de moradores abrangidos.

**Conclusão**

Contrariamente aos projetos de programação anteriormente desenvolvidos, foi-nos requerido que modelássemos o problema apresentado. O objetivo principal desta primeira fase foi ajudar-nos a compreender a realidade do projeto, de maneira a melhor entendermos a sua estrutura essencial e desta maneira planear de forma rigorosa e prevenir possíveis riscos do código a implementar.

Para tal, elaborámos para esta parte dois tipos de diagramas: o de Modelo de Domínio e o de *Use Case*.

O Modelo de Domínio serviu para planear as várias entidades que poderemos vir a precisar e a relação entre elas. Para a situação da criação de sistema de partilha de despesas presumimos que não seria algo bastante extenso, mas, no entanto, algo detalhado, o que eventualmente se acabou por verificar.

Depois partimos para a elaboração do diagrama dos *Use Case*. Aqui tentámos prever o comportamento do programa às diferentes ações de um ator. Supôs-se inicialmente que pouco diferenciaria a quantidade de ações exclusivas a cada tipo de ator, suposição que mais tarde passou a ser descartada após discussão em grupo, levando à conclusão da necessidade de três tipos de atores, cada um com tarefas particulares.

Encontrámos algumas dificuldades ao longo desta primeira fase de planeamento. A maior destas terá sido definir o que pretendíamos para o nosso programa, nomeadamente a forma como teria que dividir as despesas. Foi aqui decidimos ter categorias diferentes de despesa (dentro da extraordinária e recorrente). Também tivemos dúvidas sobre quem deveria colocar uma despesa no sistema. Por isso é que criámos a entidade Administrador, que pode ser um morador ou o senhorio. O morador apenas coloca despesas extraordinárias, enquanto que o senhorio poderá acrescentar despesas extraordinárias ou recorrentes.

Em modo de conclusão, embora esta parte inicial seja longa e trabalhosa, ser-nos-á muito útil na implementação do código uma vez que já estará tudo estruturado e documentado.