

Desenvolvimento de Sistemas de Software

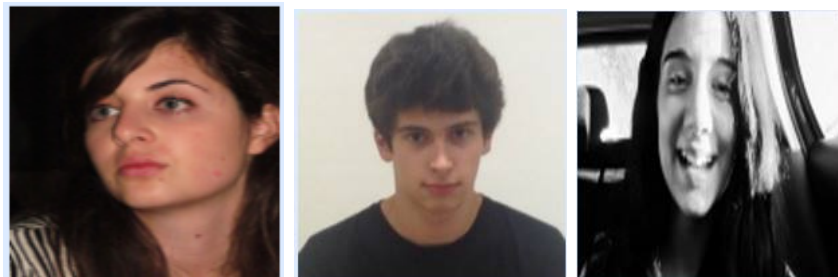
Sistema de Gestão de Despesas

Adriana Eliana Fernandes Pereira A67662

André Rodrigues Freitas A74619

Sofia Manuela Gomes de Carvalho A76658

31 de Dezembro de 2016



Conteúdo

1	Introdução	5
2	Desenvolvimento	6
2.1	Modelo de Domínio	7
2.2	Modelo de Use Case	9
2.2.1	Autenticação	11
2.2.2	AcrescentarDespesas	12
2.2.3	PagarDespesa	12
2.2.4	ConsultarDespesas	12
2.2.5	ConsultarPagamentosEfetuados	14
2.2.6	AcrescentaMorador	15
2.2.7	RemoveMorador	16
2.3	Máquina de Estado - Morador	17
2.4	Máquina de Estado - Senhorio	18
2.5	Diagramas de Sequência	20
2.5.1	Autenticar	20
2.5.2	AcrescentarDespesas	21

2.5.3	ConsultarDespesas	22
2.5.4	ConsultarPagamentosEfetuados	23
2.5.5	PagarDespesa	24
2.5.6	AcrescentarMorador	25
2.5.7	ConsultarDespesas	26
2.5.8	ConsultarPagamentosEfetuados	27
2.5.9	RemoverMorador	28
2.6	Diagramas de Sequência com Subsistemas	29
2.6.1	ConsultarDespesas	29
2.6.2	ConsultarDespesas	30
2.6.3	ConsultarPagamentosEfetuados	31
2.6.4	AcrescentarDespesas	32
2.6.5	AcrescentarMorador	33
2.6.6	Autenticar	34
2.6.7	PagarDespesa	35
2.6.8	RemoveMorador	36
2.7	Diagramas de Sequência - Implementação	37

2.7.1	Autenticar	37
2.7.2	AcrescentarDespesas	38
2.7.3	ConsultarDespesas	39
2.7.4	PagarDespesa	40
2.7.5	AcrescentarMorador	41
2.8	Diagrama de Package	42
2.9	Diagrama de Classe	43
2.10	Diagrama de Classe com DAO's	44
2.11	Diagrama de Classe com Métodos	45
2.12	Diagrama de Instalação	46
3	Conclusão	47

1 Introdução

Este trabalho pretende desenvolver um sistema de suporte à partilha de despesas num apartamento. A aplicação desenvolvida suporta o registo das várias despesas e da sua gestão a nível de pagamentos por parte de cada um dos moradores registados nesse apartamento. Cada morador tem a sua própria conta.

Este sistema tem em conta que moradores habitam o apartamento num determinado momento, tendo apenas estes permissão para fazer operações e visualizar as despesas relativas ao apartamento em causa. Para poder fazer qualquer tipo de operação neste sistema, o morador precisa de estar autenticado. Os dados para autenticação num certo apartamento (nome de utilizador e palavra-passe) são fornecidos pelo senhorio, aquando da conclusão do acordo de arrendamento.

2 Desenvolvimento

No nosso sistema, o senhorio é o único autorizado a remover um morador, ou seja, apenas este pode remover uma conta associada a um morador de um apartamento, apartamento este que o morador deixou de habitar. Cada senhorio pode, assim, estar associado a vários apartamentos mas cada morador só pode estar associado a uma e uma só habitação de cada vez.

Da mesma forma que o senhorio pode remover moradores, este pode também acrescentar moradores a um dado apartamento.

A consulta de despesas pode ser efetuada pelo senhorio e pelos moradores. Assim, um morador pode ver apenas as suas despesas e um senhorio pode ver todas as despesas de todos os moradores num determinado apartamento.

A consulta dos pagamentos efetuados funciona de forma semelhante à consulta de despesas, sendo que aqui apenas se vêem as despesas já pagas.

Da parte do morador, este tem de se autenticar para fazer qualquer operação na aplicação. Ele tem a possibilidade de acrescentar despesas ao sistema, sejam estas recorrentes (água, luz, eletricidade, gás, renda e condomínio) ou extraordinárias (por exemplo: obras no apartamento ou compras de mobiliário, entre outras).

Outra funcionalidade prevista é pagar despesas. Também aqui o morador terá de indicar o tipo de despesa e a sua modalidade de pagamento.

2.1 Modelo de Domínio

Para o Modelo De Domínio criamos várias entidades:

- Morador: esta entidade corresponde a um morador que habita o apartamento alugado;
- ExMorador: corresponde a um morador que já não habita o apartamento;
- Senhorio: o senhorio é o proprietário do apartamento, sendo que cada senhorio pode possuir um ou mais apartamentos;
- Apartamento: é ao qual se vão associar despesas, moradores e pagamentos.
- Despesa: esta entidade corresponde às despesas;
- DespesaRecorrente: vai corresponder às despesas pagas todos os meses tais como eletricidade, água, gás, entre outras. Estas poderão variar de apartamento para apartamento visto que em alguns poderão haver, por exemplo, despesas com pacotes de Internet e televisão que poderão não haver noutros.
- DespesaExtra: são o tipo de despesas extraordinárias, aquelas despesas que não constam do contrato feito. Podem resultar da necessidade de reparar algo no apartamento, entre outros;
- Pagamento: esta entidade corresponde aos pagamentos;
- Modalidade de Pagamento: vai corresponder ao modo como o pagamento pode ser feito: se por multibanco ou via internet.

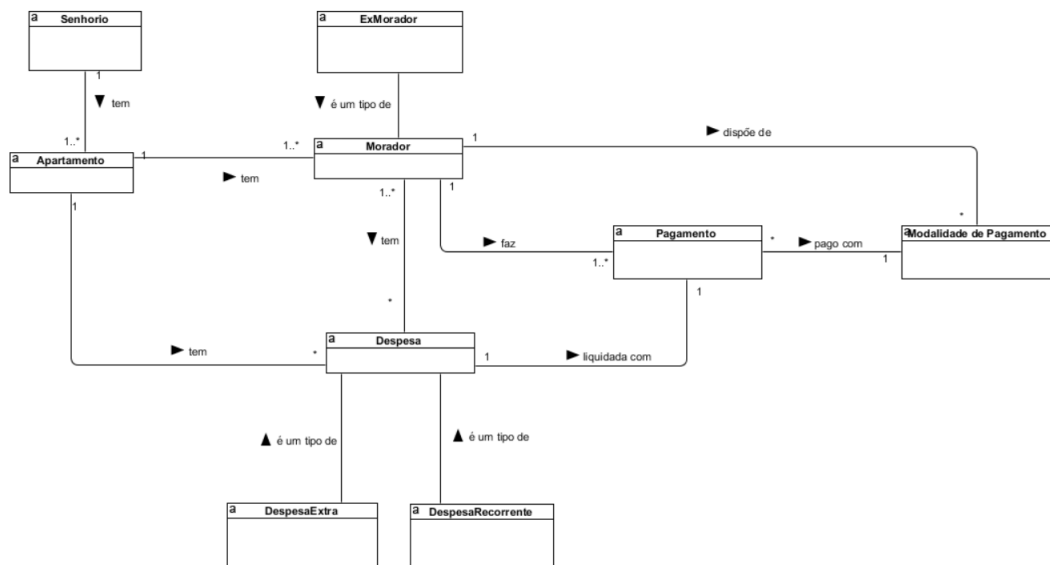


Figura 1: Modelo De Domínio

O Modelo de Domínio foi agrupado em subsistemas, sendo eles:

- SS Utilizadores;
- SS Apartamentos;
- SS Despesas;
- SS Pagamentos.

No SubSistema Utilizadores foram agrupadas as entidades relativas aos utilizadores (sendo este Morador ou Senhorio), no SS Apartamentos está a entidade Apartamento. No SS Despesas, estão as entidades Despesa, DespesaRecorrente e DespesaExtra, ou seja, neste subsistema encontram-se as entidades relativas a despesas. No último subsistema encontram-se as entidades relacionadas com Pagamentos.

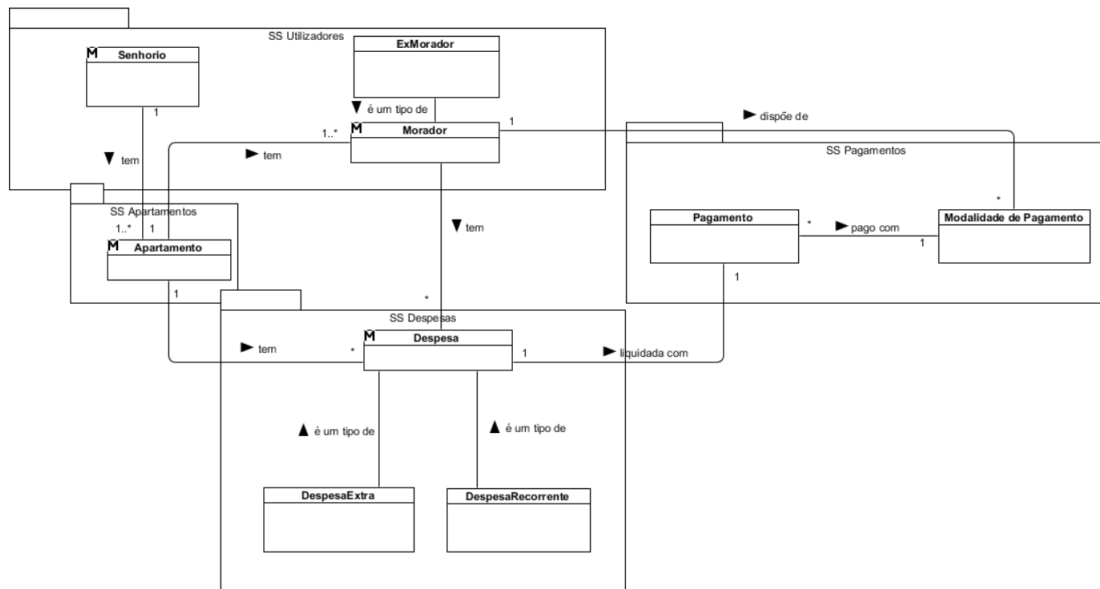


Figura 2: Modelo De Domínio com SubSistemas

2.2 Modelo de Use Case

Na definição de requisitos funcionais, identificamos dois atores: o Morador e o Senhorio, pois são aqueles que vão interagir com o sistema. Já na identificação dos Use Case, definimos sete Use Case, sendo dois deles comuns aos dois atores que vão interagir com o sistema.

Os comportamentos do sistema, da parte do Morador, são:

- Autenticação;
- AcrescentarDespesas;
- ConsultarDespesas;
- ConsultarPagamentosEfetuados;
- PagarDespesa.

Da parte do Senhorio, o sistema permite:

- AcrescentaMorador;
- ConsultarDespesas;
- ConsultarPagamentosEfetuados;
- RemoveMorador.

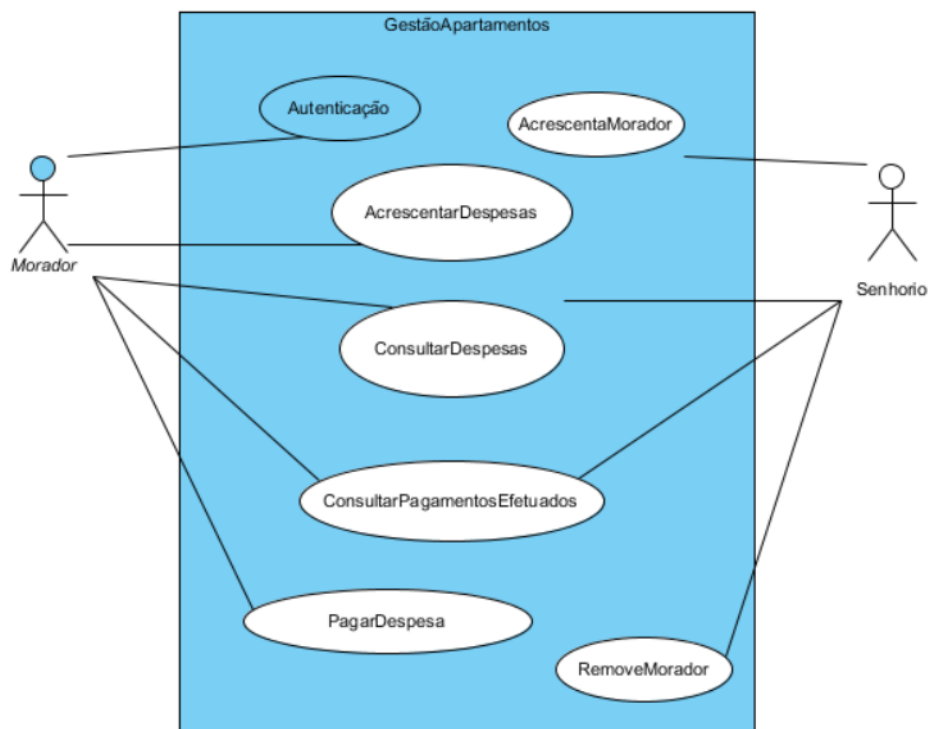


Figura 3: Diagrama de Use Case

2.2.1 Autenticação

Este Use Case vai permitir ao Morador fazer a sua autenticação no sistema, para assim poder ter acesso às suas contas, ao pagamentos efetuados e às despesas por pagar, entre outros. Para tal, é necessário que não esteja ainda nenhum Morador autenticado. Assim, o Morador escreve o seu email e a respetiva password e o sistema valida o acesso e apresenta as várias opções disponíveis. Caso o email ou a password estejam incorretas, é lançado um aviso a dizer que estas estão incorretas e volta a pedir o email e a password.

Autenticação		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Main		
Super Use Case		
Author		
Date		
Brief Description Morador autentica-se na aplicação		
Preconditions Nenhum morador autenticado		
Post-conditions Morador fica autenticado		
Flow of Events	Actor Input	
	System Response	
	1	Apresenta email e password
	2	Valida acesso
Comp. Alternative 1 (email ou password inválida) (passo 2)	Actor Input	
	System Response	
	1	Avisa sobre email ou password inválida
	2	Regressa ao passo 1

Figura 4: Especificação Tabular do Use Case Autenticação

2.2.2 AcrescentarDespesas

AcrescentarDespesas exige que o Morador esteja autenticado. Quando tal acontece, este indica o valor da despesa, a sua data de emissão e a data limite de pagamento e o sistema regista esta despesa.

AcrescentarDespesas		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Super Use Case		
Author		
Date		
Brief Description	Acrescenta despesas de um certo tipo	
Preconditions	Utilizador autenticado	
Post-conditions	Despesa adicionada	
Flow of Events	Actor Input	System Response
1	Indica valor da despesa, data de emissão e data limite de pagamento	
2		Regista despesa

Figura 5: Especificação Tabular do Use Case AcrescentarDespesas

2.2.3 PagarDespesa

Este Use Case exige que o Morador esteja autenticado, sendo que o sistema vai mostrar o tipo de despesas e o ator seleciona de seguida o tipo de despesa. Se não tiver despesas desse tipo por pagar, o sistema volta a mostrar os tipos de despesas. Caso contrário, o ator pode escolher a despesa que quer pagar e o sistema propõe-lhe duas modalidades de pagamento: por multibanco ou pela internet. Consoante escolha do morador, irá gerar referências bancárias ou uma senha única de pagamento, respetivamente, para que o pagamento da despesa possa ser feito.

2.2.4 ConsultarDespesas

Este Use Case é comum aos dois atores do sistema. Permite consultar a lista de um dado tipo de despesas. Para tal, o ator indica que quer consultar a lista de despesas e o sistema fornece os tipos de despesas. De seguida, o ator seleciona o tipo de despesa que quer consultar e é apresentada uma lista de despesas do tipo selecionado. Se, ao selecionar um tipo que não tem

Super Use Case		
Author		
Date		
Brief Description		
Preconditions		
Post-conditions		
Flow of Events		Actor Input
		System Response
	1	Mostra tipos de contas
	2	Seleciona tipo de contas
	3	Mostra contas por pagar
	4	Seleciona contas que quer pagar
	5	Mostra modalidades de pagamento
	6	Seleciona pagar com multibanco
	7	Gera referência bancária
Comp.Alternativo1 (passo 3) [sem contas daquele tipo por pagar]		
		Actor Input
		System Response
	1	Regressa a 1
Comp.Alternativo2 (passo 7) [escolhe outra modalidade de pagamento]		
		Actor Input
		System Response
	1	Gera senha única de pagamento

Figura 6: Especificação Tabular do Use Case PagarDespesas

nenhuma despesa, o sistema indica que não há nenhuma despesa desse tipo na lista e volta a fornecer os tipos de despesa para, quem sabe, nova escolha de tipo.

ConsultarDespesas		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Agency FB 8		
Main		
Super Use Case		
Author		
Date		
Brief Description		
Preconditions		
Post-conditions		
Flow of Events		Actor Input
		System Response
	1	Indica que quer consultar a lista de despesas
	2	Fornecer os tipos de despesas
	3	Seleciona o tipo de despesas que quer consultar
	4	Mostra a lista de despesas desse tipo
Comp.Alternativo (passo 4) [não há lista de despesas]		
		Actor Input
		System Response
	1	Indica que não há nenhuma despesa na lista
	2	Regressa a 2

Figura 7: Especificação Tabular do Use Case ConsultarDespesas

2.2.5 ConsultarPagamentosEfetuados

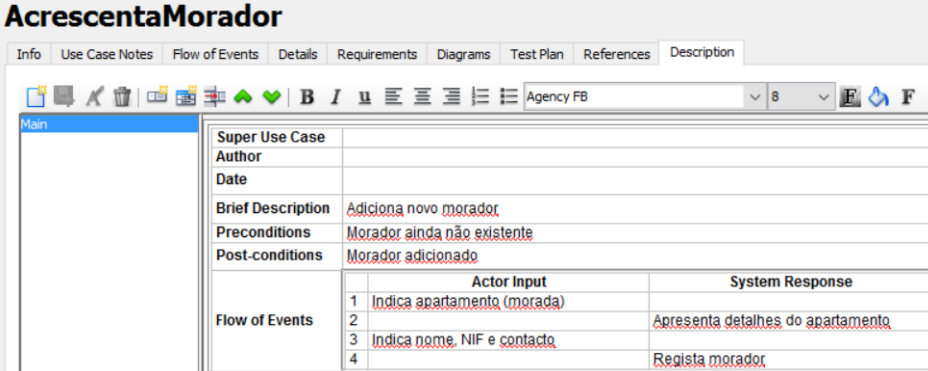
É outro Use Case comum aos dois atores. É necessário ou ser um Morador autenticado ou ser Senhorio para poder consultar os pagamentos já efetuados. O ator indica que quer consultar a lista de pagamentos efetuados e o sistema devolve o tipo de despesas para o ator selecionar. Este seleciona o tipo de pagamento que quer consultar e é devolvida pelo sistema a respetiva lista de pagamentos. Contudo, o sistema pode divulgar uma mensagem sobre o facto de não existir nenhum pagamento efetuado do tipo selecionado e volta a apresentar os tipos de despesas.

ConsultarPagamentosEfetuados		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Agency FB 8		
Main	Super Use Case	
	Author	
	Date	
	Brief Description	
	Preconditions	
	Post-conditions	
	Flow of Events	
	Comp.Alternativo (passo 5)	
	[não há lista de pagamentos efetuados]	
1		Indica que quer consultar a lista de pagamentos já efetuados
2		Fornecer os tipos de despesas
3		Selecionar o tipo de pagamentos efetuados que quer consultar
4		Mostrar a lista de pagamentos efetuados desse tipo
1		Indicar que não há nenhum pagamento efetuado do tipo selecionado na lista
2		Regressar a 3

Figura 8: Especificação Tabular do Use Case ConsultarPagamentosEfetuados

2.2.6 AcrescentaMorador

Este Use Case permite adicionar um morador ainda não existente. O Senhorio indica em que apartamento quer acrescentar o novo morador e o sistema apresenta os detalhes do apartamento. De seguida, o Senhorio fornece o nome, o NIF e o contacto do novo morador e o sistema regista-o.



AcrescentaMorador		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Main		
Super Use Case		
Author		
Date		
Brief Description Adiciona novo morador		
Preconditions Morador ainda não existente		
Post-conditions Morador adicionado		
Flow of Events		
	Actor Input	System Response
1	Indica apartamento (morada)	
2		Apresenta detalhes do apartamento
3	Indica nome, NIF e contacto	
4		Regista morador

Figura 9: Especificação Tabular do Use Case AcrescentaMorador

2.2.7 RemoveMorador

Apenas o Senhorio pode aceder a este Use Case que remove um morador. Para tal, indica o apartamento sobre o qual quer remover o morador e o sistema apresenta os detalhes do apartamento. O Senhorio indica o NIF do morador a remover e o sistema remove-o.

RemoveMorador		
Info Use Case Notes Flow of Events Details Requirements Diagrams Test Plan References Description		
Agency FB 8		
Main		
Super Use Case		
Author		
Date		
Brief Description	Remove um <u>morador</u>	
Preconditions	Feito pelo <u>Senhorio</u>	
Post-conditions	<u>Morador removido</u>	
Flow of Events		
	Actor Input	System Response
	1 <u>Indica apartamento (morada)</u>	
	2	<u>Apresenta detalhes do apartamento</u>
	3 <u>Indica NIF</u>	
	4	<u>Remove <u>morador</u> associada ao NIF escolhido</u>

Figura 10: Especificação Tabular do Use Case RemoveMorador

2.3 Máquina de Estado - Morador

A Máquina de Estado apresentada permite modelar o comportamento do sistema de suporte à partilha de despesas num apartamento de forma global, do ponto de vista do Morador.

Apresentam-se modelados todos os estados possíveis pelos quais o sistema pode atravessar em resposta aos eventos que podem ocorrer. Modelamos o comportamento do sistema como um todo, em particular a interface com o utilizador, mais especificamente, neste caso, com o Morador.

O Morador abre a aplicação e enquanto não se autenticar não tem permissões para poder utilizar a aplicação. Para se autenticar, o utilizador faz login fornecendo o seu nome de utilizador e a password. Se algum destes parâmetros estiver incorreto ou não constar na Base de Dados, o utilizador continua sem permissões, podendo efetuar, de novo, a sua autenticação.

A partir do momento em que se autentica, é-lhe apresentada uma janela principal onde este pode selecionar "Despesas" ou "Pagamentos".

Selecionando "Pagamentos", é apresentada no ecrã a lista com os pagamentos já efetuados e o utilizador pode selecionar especificamente qual o pagamento que pretende analisar ou apenas cancelar, voltando à janela principal. Aquando desta escolha, o Morador pode visualizar as informações do respetivo pagamento e pode apenas selecionar "ok", voltando à janela principal.

Se na janela principal selecionar "Despesas", tem a opção de aceder à lista de despesas já inseridas, inserir uma nova despesa ou cancelar, regressando à janela principal. Se selecionar a lista de despesas já inseridas, esta lista é-lhe apresentada no ecrã e pode selecionar uma dada despesa e obter, assim, informações sobre essa despesa. Se pretender inserir uma nova despesa, seleciona o tipo de despesa pretendido e insere a nova despesa. Pode sempre cancelar a nova inserção ou confirmar a inserção, carregando "ok".

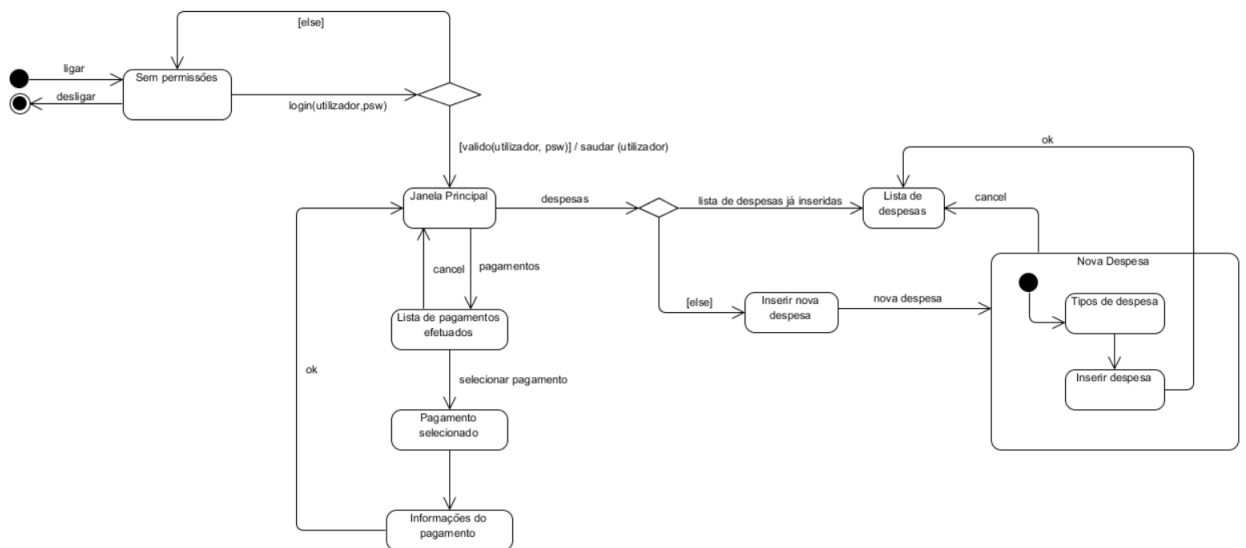


Figura 11: Máquina de Estado referente ao ponto de vista do Morador

2.4 Máquina de Estado - Senhorio

Esta Máquina de Estado modela o comportamento do sistema do ponto de vista do Senhorio. Este é o único utilizador que não precisa de se autenticar para poder aceder às funcionalidades da aplicação. Este utilizador abre a aplicação e pode seleccionar uma das quatro opções fornecidas na janela principal: "Acrescentar novo morador", "Remover um morador", "Despesas" e "Pagamentos".

Selecionando a primeira opção ("Acrescentar novo morador"), insere um novo morador clicando "ok" para confirmar a sua inserção ou cancela a opção recorrendo ao "cancel" e é retornado para a janela principal. A opção "Remover um morador" é semelhante a esta, diferenciando apenas que remove um morador e clicando "ok" confirma a sua atividade ou recorre ao "cancel" para cancelar esta operação de remover um morador do apartamento em questão.

Selecionando "Pagamentos", é apresentada no ecrã a lista com os pagamentos já efetuados e o senhorio pode seleccionar especificamente qual o pagamento que pretende analisar ou apenas cancelar, voltando à janela principal. Aquando desta escolha, o são-lhe apresentadas as informações do respetivo pagamento e pode apenas seleccionar "ok", voltando de novo à janela

principal.

Se na janela principal seleccionar "Despesas", tem a opção de aceder à lista de despesas já inseridas ou cancelar, regressando à janela principal. Se seleccionar a lista de despesas já inseridas, esta lista é-lhe apresentada no ecrã e pode seleccionar uma dada despesa e obter, assim, informações sobre essa despesa. Para regressar à janela principal, tem de carregar na única opção disponível ("ok") em qualquer uma destas duas opções seleccionadas após ter escolhido "Despesas".

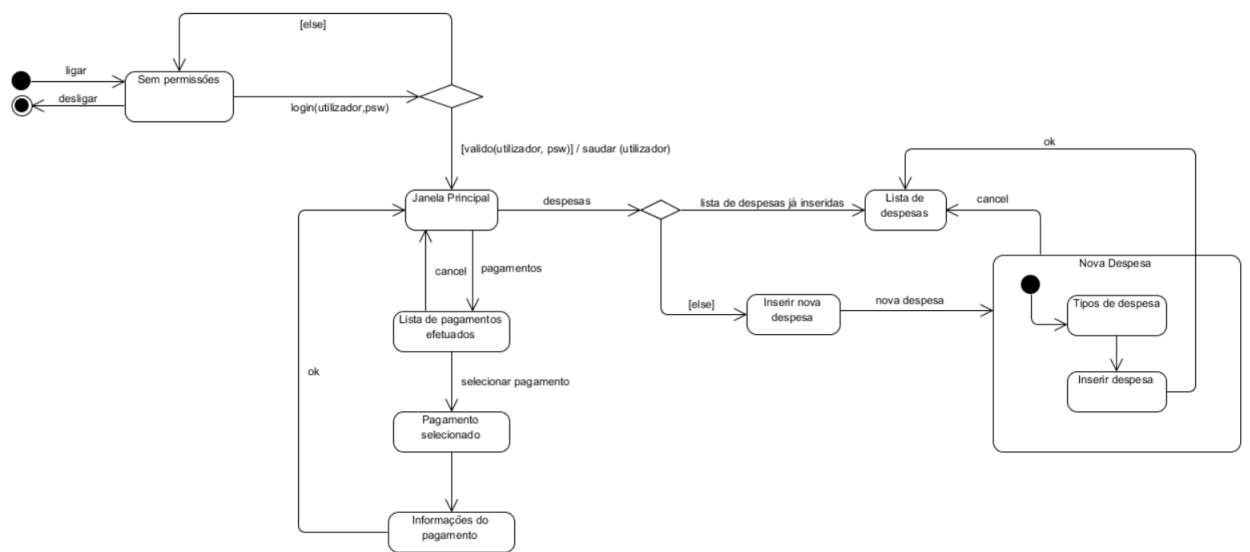


Figura 12: Máquina de Estado referente ao ponto de vista do Senhorio

2.5 Diagramas de Sequência

2.5.1 Autenticar

Este Diagrama de Sequência representa as interações entre o Morador e o SGD. Primeiramente, o Morador envia uma mensagem onde indica o seu email e a sua password para se autenticar. O Sistema de Gestão de Despesas verifica esse Morador, procurando se o email fornecido já se encontra registado. Como o email pode já ter sido registado, ou não, necessitamos de criar um fragmento alternativo que vai expressar o fluxo condicional de enviar uma mensagem de resposta em que diz que é impossível autenticar, caso o email ainda não tenha sido registado. O outro fluxo condicional indica a confirmação da autenticação, ou seja, o email já se encontra registado.

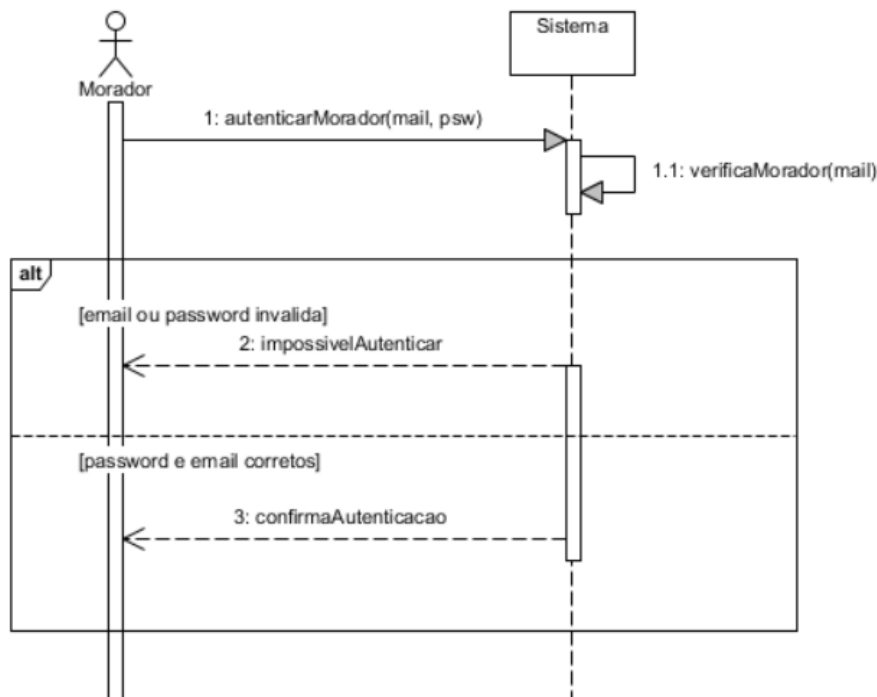


Figura 13: Diagrama de Sequência que ilustra a autenticação do Morador

2.5.2 AcrescentarDespesas

Na figura 14 encontra-se o Diagrama de Sequência que permite analisar toda a troca de mensagens que ocorre quando um Morador pretende acrescentar uma nova despesa. Inicialmente, o Morador envia ao Sistema de Gestão de Despesas (SGD) e este acrescenta a despesa, retornando uma mensagem onde confirma a inserção da nova despesa.

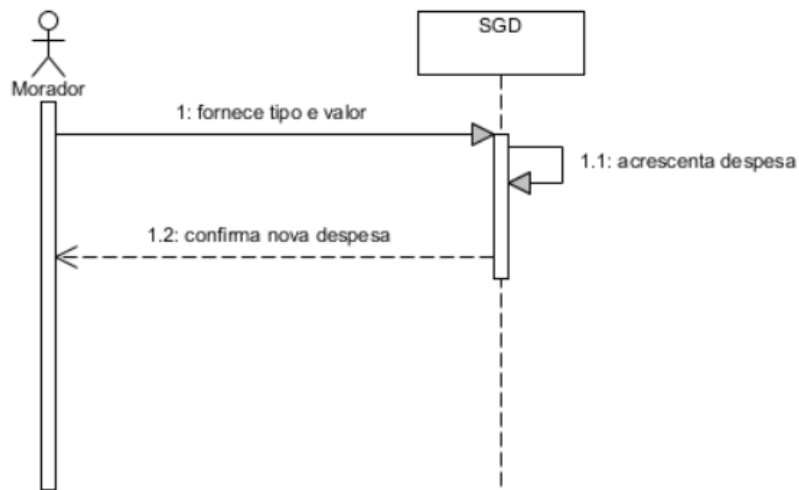


Figura 14: Diagrama de Sequência AcrescentarDespesas

2.5.3 ConsultarDespesas

Neste Diagrama de Sequência, o Morador envia uma mensagem do SGD com o tipo da despesa que pretende consultar e o sistema procura as despesas desse tipo. Caso não haja despesas desse tipo, retorna uma mensagem onde indica que não existe nenhuma despesa. Caso contrário, a mensagem retornada devolve as despesas do tipo pretendido. Para implementar estes dois casos, foi usado um fragmento alternativo.

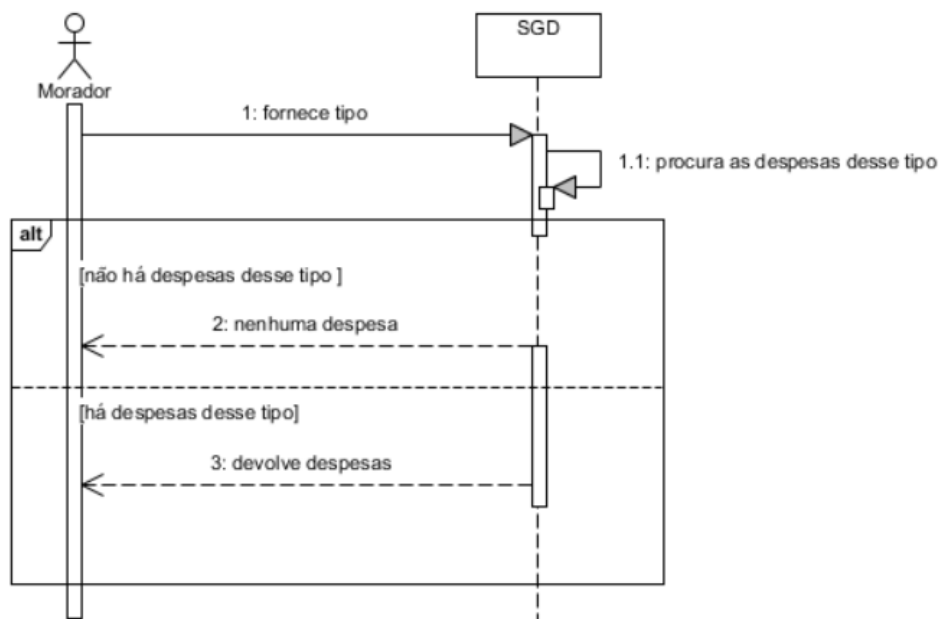


Figura 15: Diagrama de Sequência ConsultarDespesas(Morador)

2.5.4 ConsultarPagamentosEfetuados

Na figura 16, o Morador fornece o tipo de pagamentos que está à procura ao sistema e este por sua vez irá procurar esse tipo de pagamentos. Caso existam pagamentos desse tipo, irá devolver todos os pagamentos desse tipo ao Morador. Em alternativa, pode acontecer de não existirem pagamentos desse tipo e, por isso, o SGD irá indicar isso mesmo ao Morador.

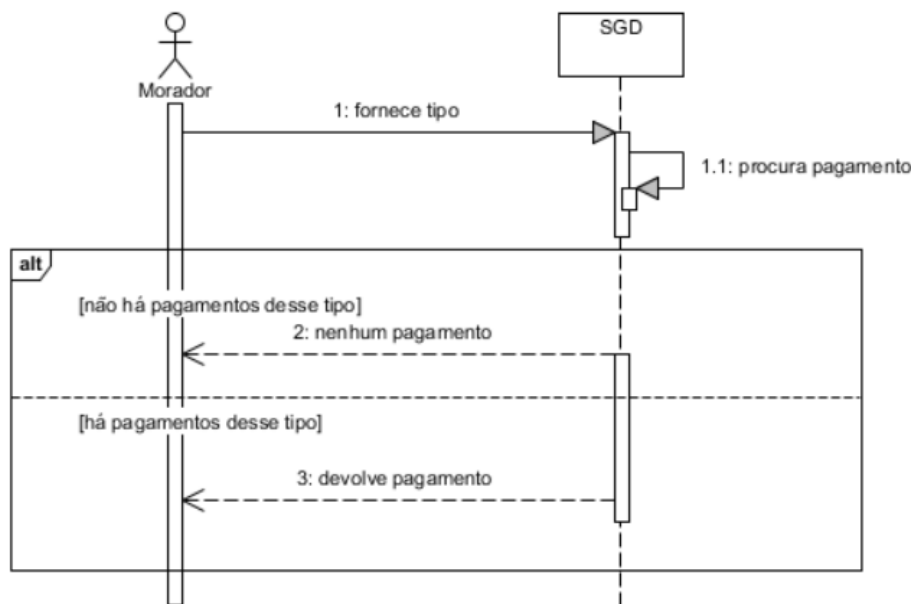


Figura 16: Diagrama de Sequência ConsultarPagamentosEfetuados(Morador)

2.5.5 PagarDespesa

Para pagar uma despesa, o Morador fornece o id da despesa, o SGD procura a despesa respectiva a esse id e necessita de dois fragmentos alternativos: um que indica que não existe nenhuma despesa com o id dado quando esta situação se verifica e outro que, quando há despesa com o id dado, paga a despesa e envia uma mensagem ao Morador a confirmar que o pagamento foi efetuado.

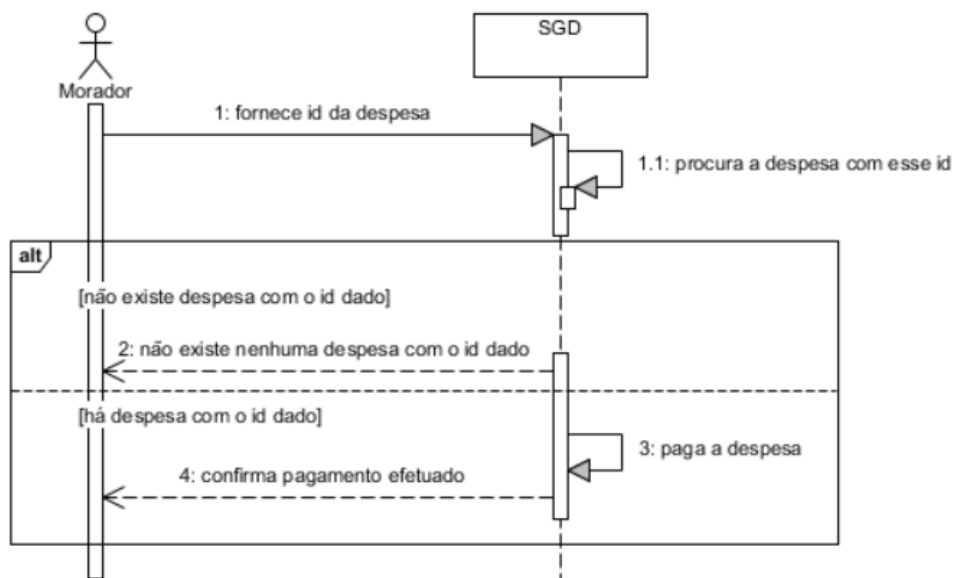


Figura 17: Diagrama de Sequência PagarDespesa

2.5.6 AcrescentarMorador

Uma das opções de que o Senhorio pode usufruir é acrescentar um novo morador. Para isso, terá de indicar o NIF do Morador que quer acrescentar que será verificado pelo SGD. Se não existir ainda no sistema, então o morador é acrescentado. Caso contrário, ou seja, quando o morador já existe, não é acrescentado.

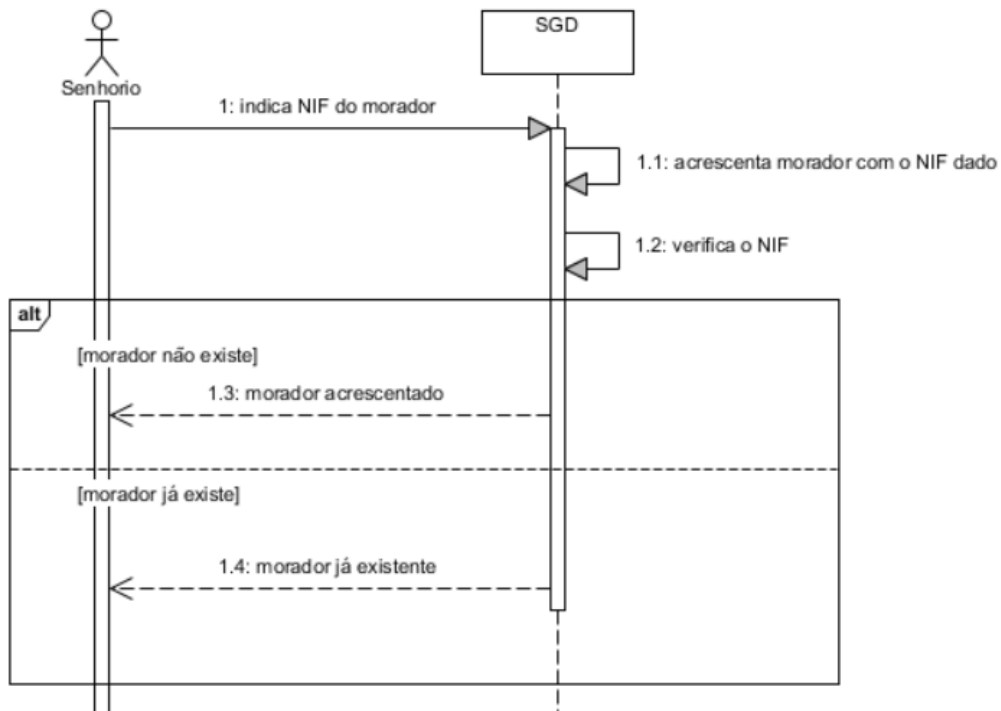


Figura 18: Diagrama de Sequência AcrescentarMorador

2.5.7 ConsultarDespesas

Na figura 19, encontra-se o Diagrama de Sequência que mostra todas as mensagens trocadas entre o Senhorio e o SGD quando este ator pretende consultar as despesas de um dado tipo. Assim, o Senhorio fornece o tipo que pretende visualizar e, deste modo, o SGD procura todas as despesas deste tipo. É necessário criar um fragmento alternativo em que um fluxo condicional não retorna nenhuma despesa se não houver despesas do tipo fornecido. Caso contrário, são devolvidas todas as despesas desse tipo.

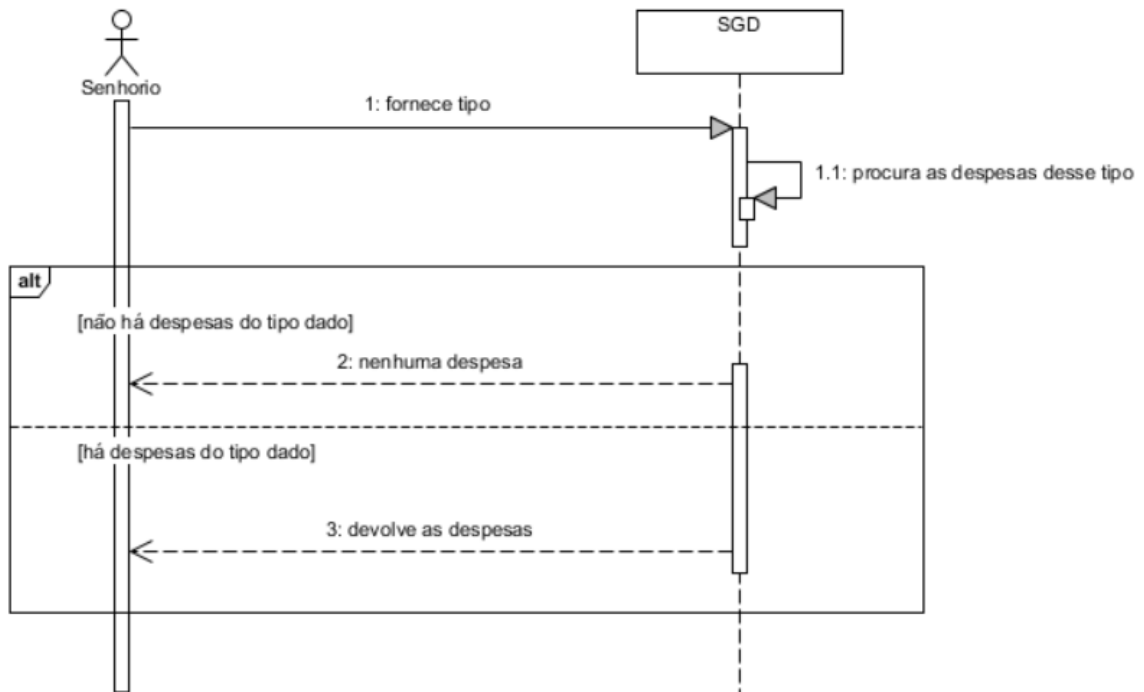


Figura 19: Diagrama de Sequência ConsultarDespesas(Senhorio)

2.5.8 ConsultarPagamentosEfetuados

O Senhorio pode também consultar os pagamentos efetuados fornecendo o tipo de pagamentos que está à procura ao sistema e este por sua vez irá procurar esse tipo de pagamentos. Caso existam pagamentos desse tipo, irá devolver todos os pagamentos desse tipo ao Senhorio. Em alternativa, pode acontecer de não existirem pagamentos desse tipo e, por isso, o SGD irá indicar isso mesmo ao Senhorio.

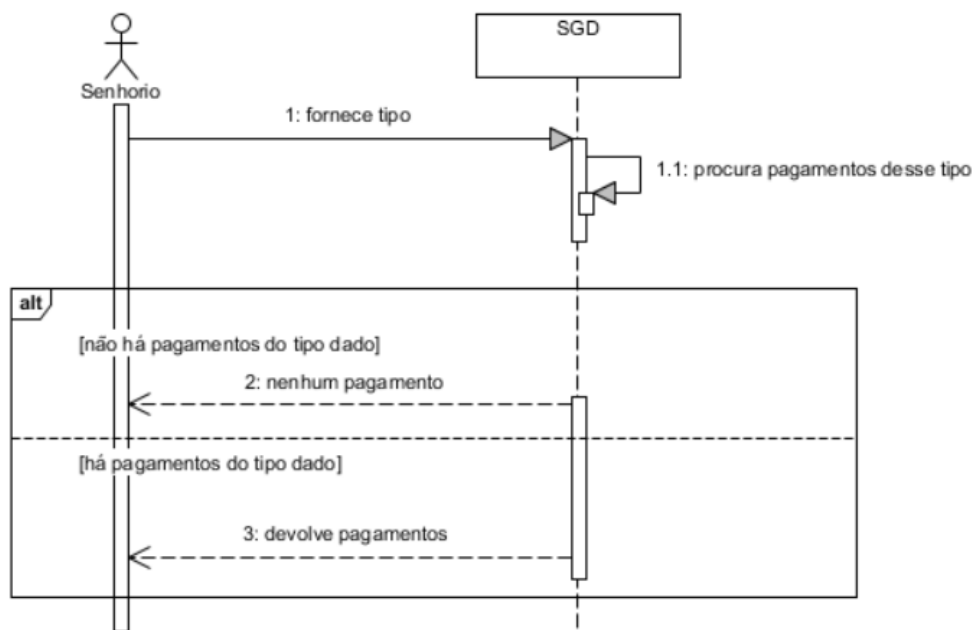


Figura 20: Diagrama de Sequência ConsultarPagamentosEfetuados(Senhorio)

2.5.9 RemoverMorador

Na figura 21, o Senhorio pode fornecer o NIF de um morador que queira remover, sendo este NIF procurado no SGD. Caso exista, o morador com esse NIF será removido. Caso contrário, não será removido nenhum morador.

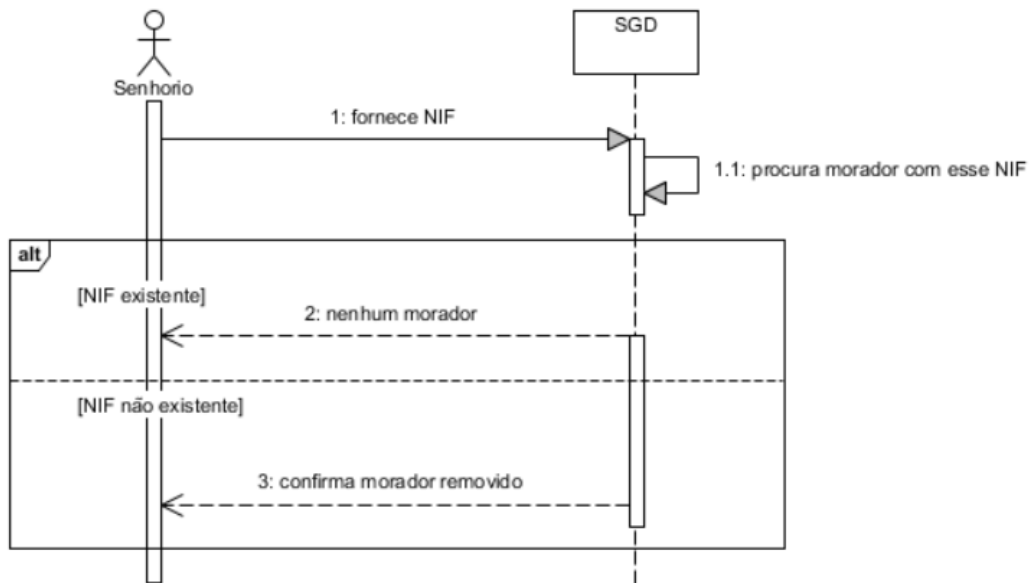


Figura 21: Diagrama de Sequência RemoverMorador

2.6 Diagramas de Sequência com Subsistemas

2.6.1 ConsultarDespesas

Este diagrama difere do apresentado na figura 22 pois este diagrama tem em consideração a agrupação das classes do Modelo de Domínio em subsistemas (figura 2).

O Morador fornece o tipo, o SGD consulta as despesas desse tipo e envia esta mensagem ao SS Despesas que irá, por sua vez, procurar as despesas desse tipo. É criado um comportamento alternativo em que o primeiro fluxo condicional é usado quando não há despesas do tipo fornecido e o SS Despesas envia uma mensagem ao SGD onde diz que não existe nenhuma despesa e o SGD envia esta mensagem ao Morador. O outro fluxo condicional é acionado quando existem despesas do tipo dado pelo Morador e essas despesas são passadas do SS Despesas para o SGD e deste são reencaminhadas para o Morador.

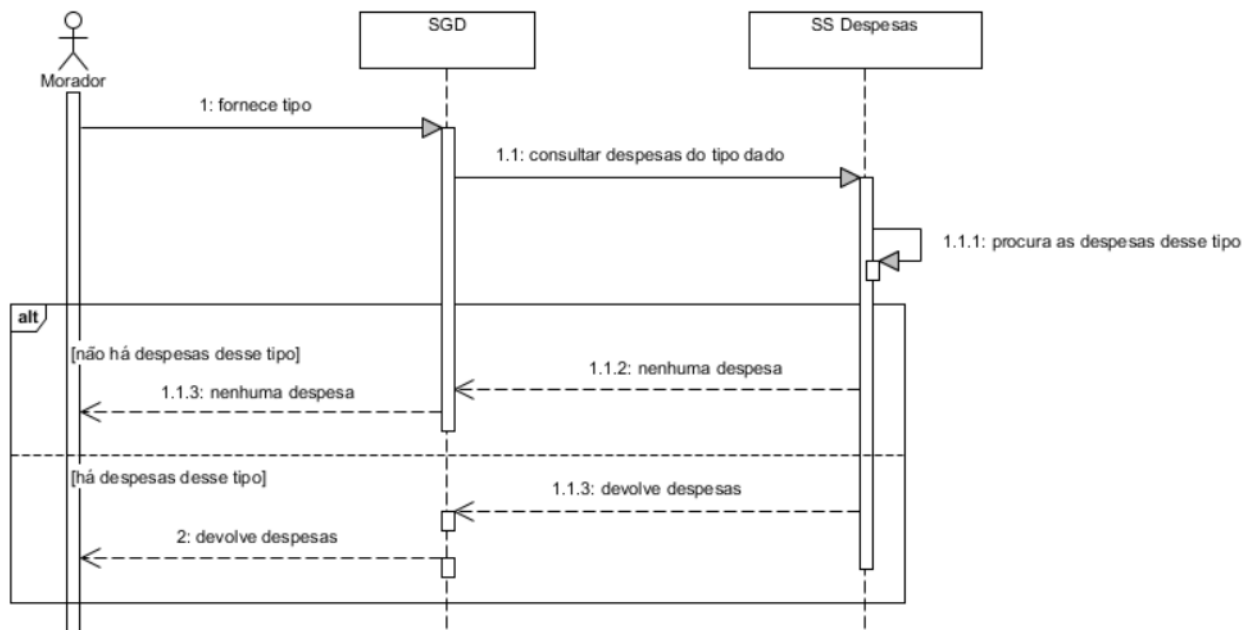


Figura 22: Diagrama de Sequência ConsultarDespesas com SubSistemas (Morador)

2.6.2 ConsultarDespesas

Neste Diagrama, o Senhorio fornece o tipo de despesa que quer consultar e o SGD consulta as despesas com esse tipo, enviando uma mensagem ao SS Despesas, e o subsistema vai procurar as despesas desse tipo. É criado um comportamento alternativo. Se não houver nenhuma despesa com esse tipo, o SS Despesas retorna uma mensagem ao SGD e este reencaminha a mesma mensagem ao Senhorio. Caso contrário, SS Despesas devolve uma mensagem com as despesas ao SGD e o SGD devolve essa mensagem ao Senhorio.

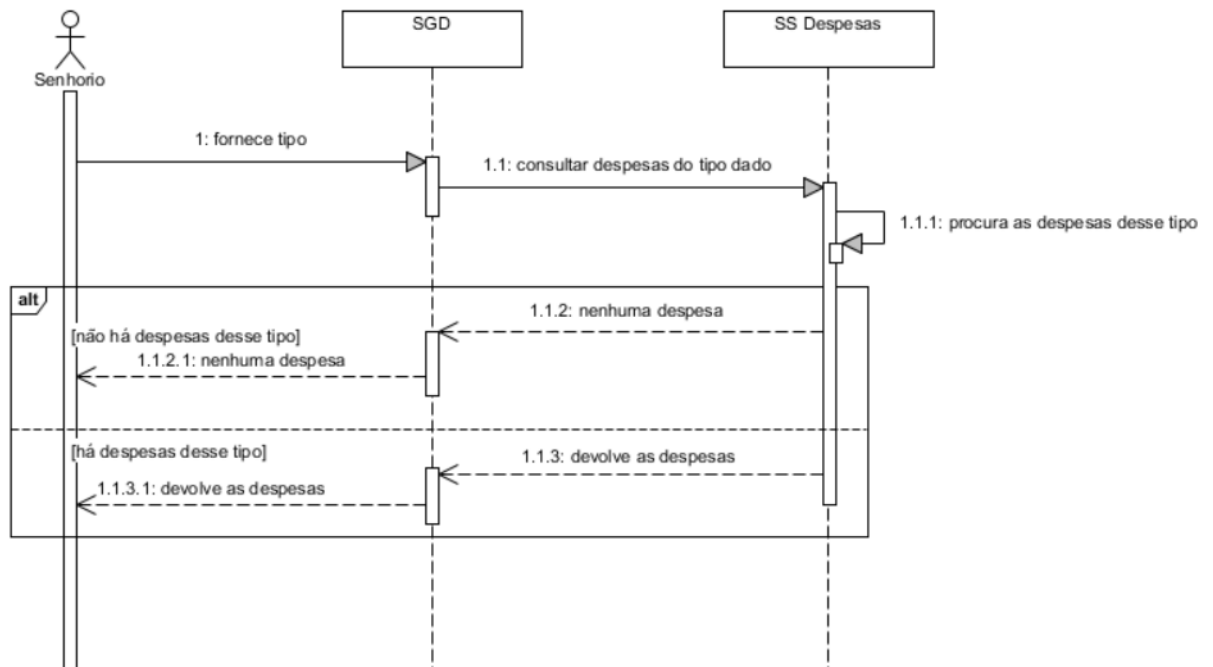


Figura 23: Diagrama de Sequência ConsultarDespesas com SubSistemas (Senhorio)

2.6.3 ConsultarPagamentosEfetuados

Este Diagrama difere do apresentado na figura 20 apenas porque o Senhorio fornece o tipo ao SGD e este indica que pretende consultar os pagamentos efetuados do tipo fornecido. No comportamento alternativo, a única diferença encontra-se apenas no facto de que as mensagens retornadas pelo SS Pagamentos são primeiramente retornadas ao SGD e deste alcançam o Senhorio.

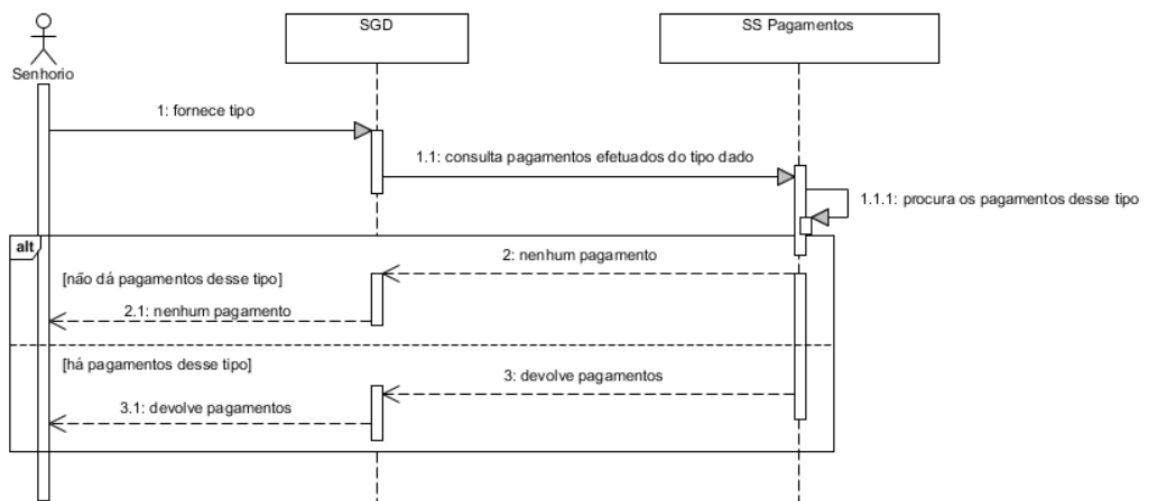


Figura 24: Diagrama de Sequência ConsultarPagamentosEfetuados com Sub-Sistemas

2.6.4 AcrescentarDespesas

No diagrama desta figura, vemos que o Morador fornece o tipo e o valor da despesa que quer acrescentar ao SGD. Este, por sua vez, vai enviar uma mensagem ao SS Despesas para que essa despesa seja acrescentada. De seguida, o subsistema das despesas irá confirmar que a despesa foi adicionada ao SGD e este fornece a informação dessa confirmação ao Morador.

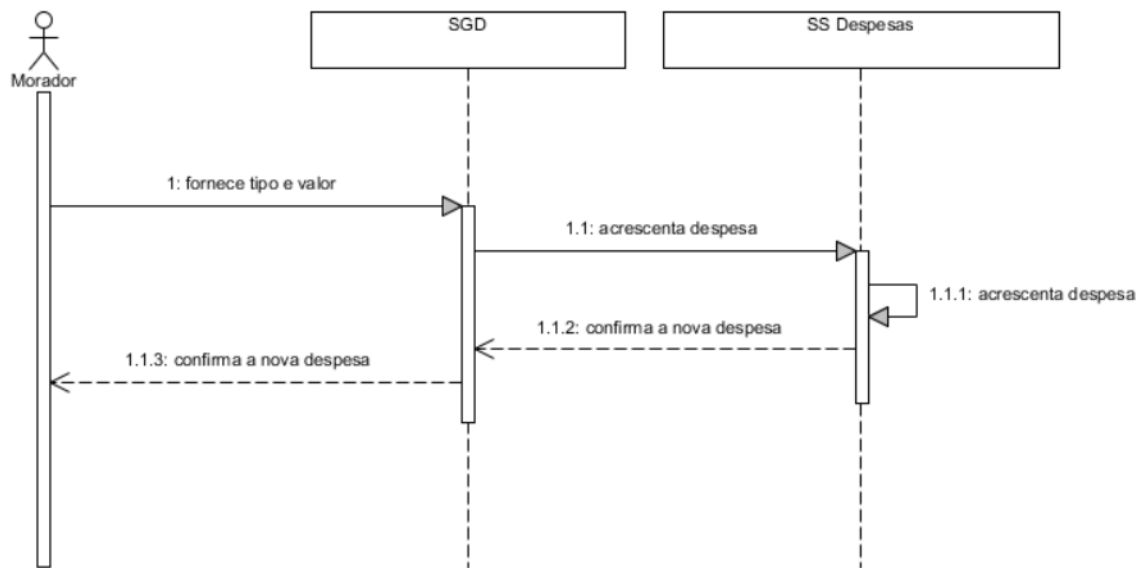


Figura 25: Diagrama de Sequência AcrescentarDespesas com SubSistemas

2.6.5 AcrescentarMorador

Para acrescentar um morador, o Senhorio terá primeiro de indicar o apartamento onde esse morador irá ficar alojado e, por isso, faz essa indicação ao SGD. O SGD terá de enviar uma mensagem ao SS Apartamentos para procurar os detalhes do apartamento em questão e este subsistema irá enviar os detalhes ao SGD que, por sua vez, os envia ao Senhorio.

De seguida, o Senhorio irá indicar os dados do morador que quer acrescentar ao SGD e este irá enviar uma mensagem ao SS Utilizadores para acrescentar esse morador. Este subsistema vai procurar esse morador e caso não exista, será acrescentado, sendo dada essa informação ao SGD que, por sua vez, a dá ao Senhorio. Caso exista, não será acrescentado o morador e será passada ao SGD a informação de que esse morador já existe, sendo de seguida dada essa informação ao Senhorio.

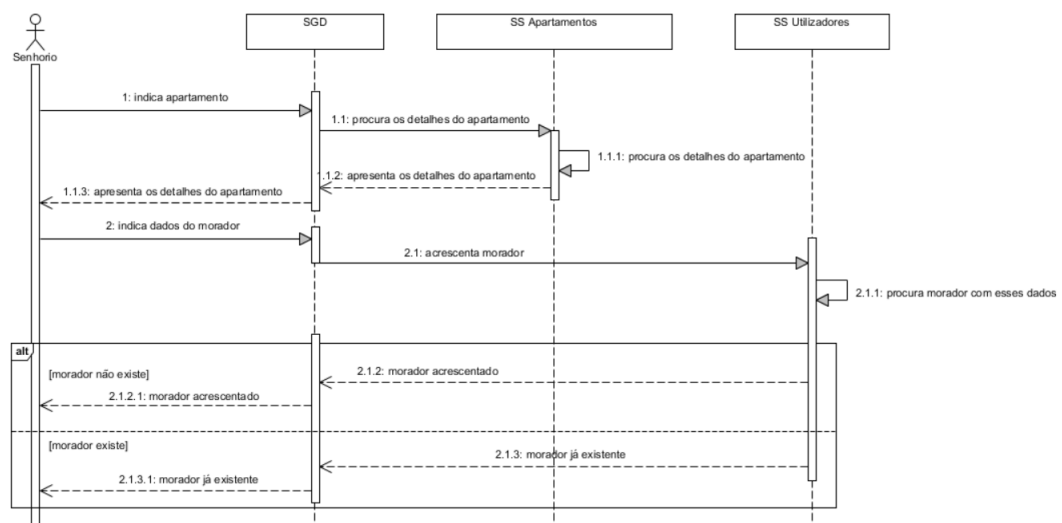


Figura 26: Diagrama de Sequência AcrescentarMorador com SubSistemas

2.6.6 Autenticar

Esta versão do Autenticar implica o envio do email e da password por parte do Morador ao SGD e este indica, de seguida, que pretende verificar se o email e a password estão já registados. O SS Utilizadores verifica o email e/ou a password e se pelo menos um destes estiver incorreto, seja porque está mal escrito ou porque ainda não foi registado, envia uma mensagem de erro para o SGD a indicar que o email ou a password são inválidos e esta mensagem é retornada ao Morador. Se nem o email nem a password estiverem errados, a mensagem que circula entre o SS Utilizadores e o SGD e de seguida entre o SGD e o Morador é a de que o email e a password são válidos e a autenticação é confirmada, respetivamente.

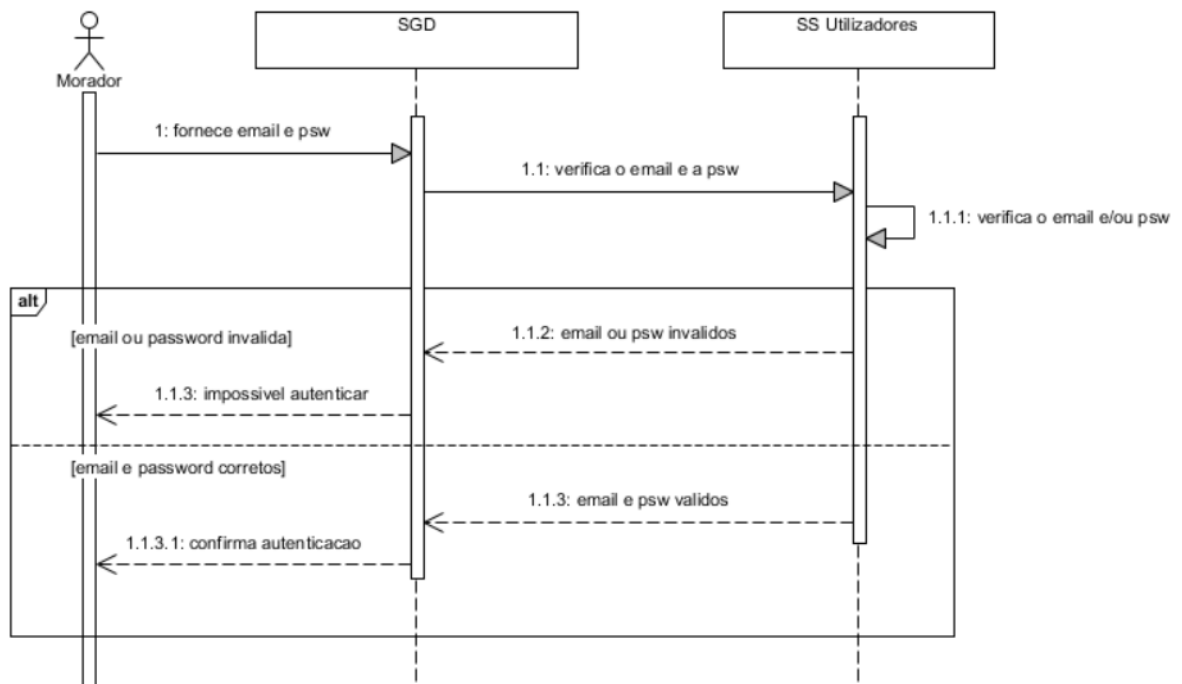


Figura 27: Diagrama de Sequência Autenticar com SubSistemas

2.6.7 PagarDespesa

O Morador envia uma mensagem ao SGD em que fornece o id da despesa e o SGD indica ao SS Despesa que deve procurar a despesa com esse id e este assim o faz. É criado um comportamento alternativo com um fluxo alternativo para o caso em que não existe nenhuma despesa com o id dado e um caso existe uma despesa com esse id. No primeiro caso, o SS DEspesas envia uma mensagem ao SGD a dizer que não existe nenhuma despesa com o id dado e é enviada uma mensagem com o mesmo conteúdo do SGD ao Morador. No outro fragmento alternativo, é enviada uma confirmação desde o SS Despesas até ao SGD a garantir que existe uma despesa com o id fornecido. Do SGD até ao SS Pagamentos é enviada uma mensagem onde aquele avisa que quer pagar a despesa e o SS Pagamentos faz o pagamento da despesa com o id passado pelo Morador. O pagamento é confirmado com o envio de uma mensagem do SS Pagamentos para o SGD e essa mensagem é transmitida do SGD para o Morador.

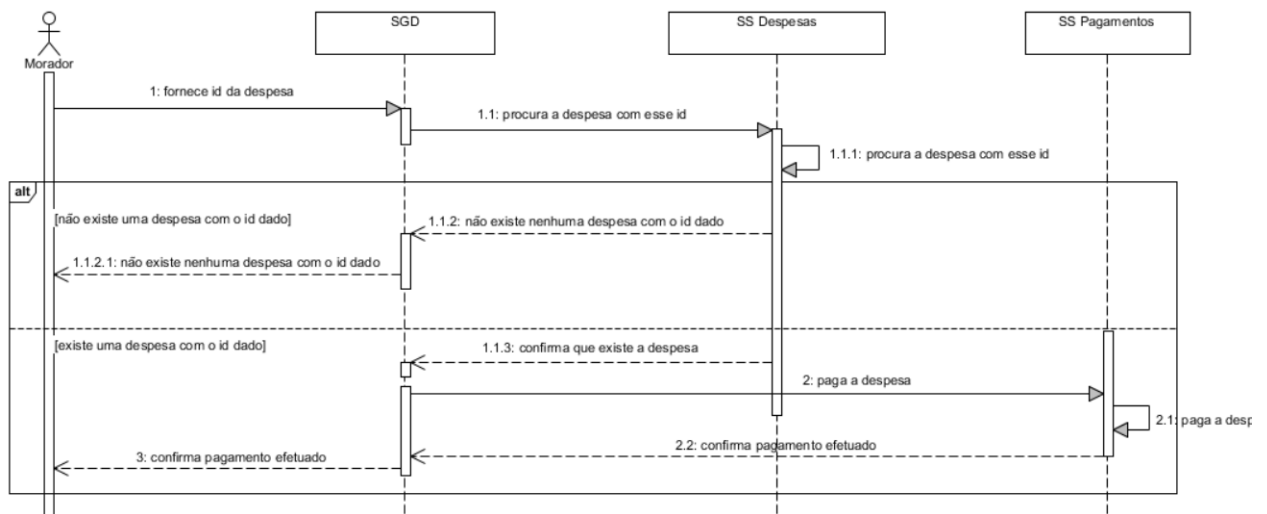


Figura 28: Diagrama de Sequência PagarDespesa com SubSistemas

2.6.8 RemoveMorador

Neste Diagrama de Sequência, depois de o Senhorio indicar o apartamento, o SGD avisa o SS Apartamentos que obtenha os detalhes do apartamento e este assim o faz, retornando uma mensagem ao SGD com os detalhes. Estes detalhes são, de seguida, enviados pelo SGD ao Senhorio. O Senhorio indica ao SGD o NIF do Morador que pretende remover do apartamento que forneceu inicialmente e esta informação é passada do SGD ao SS Utilizadores, subsistema este que procura o morador com esse NIF. Caso não exista nenhum morador com esse NIF, o SS Utilizadores envia uma mensagem para o SGD, que a irá reenviar para o Senhorio, a indicar que não existe nenhum morador naquele apartamento com o NIF fornecido. Caso contrário, a mensagem enviada do SS Utilizadores para o SGD é o facto de existir o NIF e, deste modo, o SGD indica ao SS Utilizadores que pretende remover esse morador e o SS Utilizadores procede à remoção. Quando esta termina, retorna uma mensagem ao SGD a indicar que o morador foi removido. Uma mensagem semelhante circula também entre o SGD e o Senhorio.

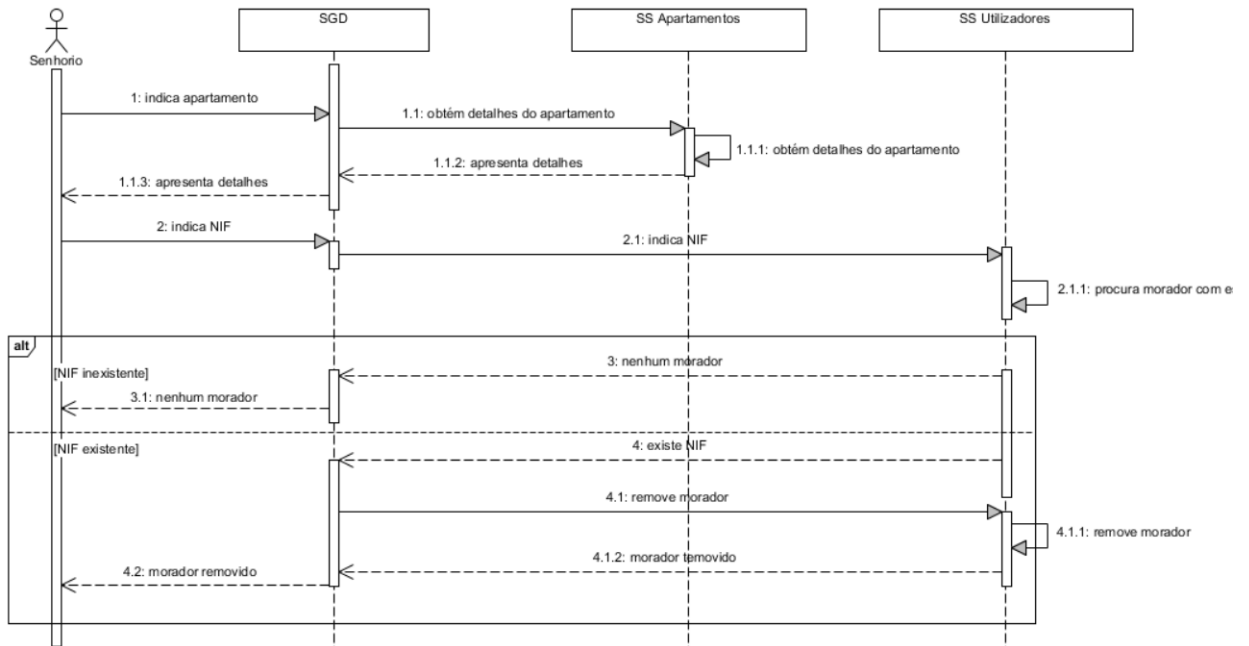


Figura 29: Diagrama de Sequência RemoveMorador com SubSistemas

2.7 Diagramas de Sequência - Implementação

2.7.1 Autenticar

Neste diagrama, podemos ver a forma como o Autenticar é implementado. Primeiro, o Morador irá fornecer o seu email e password através de uma interface (GUI). Essa interface irá invocar o método `autenticarMorador` que recebe um email e uma password e irá ao SGD. Através do SGD irá verificar se algum morador tem esse email e essa password. Caso o email ou a password sejam inválidos, ou seja, nenhum morador possui esse email e password, o GUI irá informar o Morador de que é impossível autenticar. Caso contrário, o email e a password estão corretos e a interface confirma a autenticação.

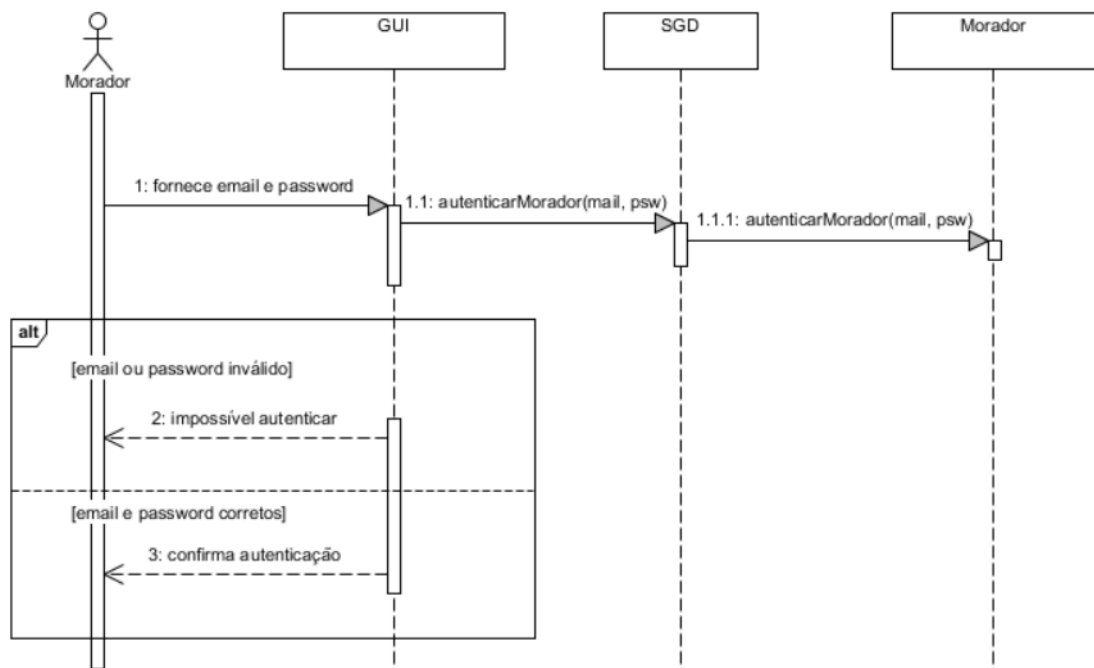


Figura 30: Diagrama de Sequência Autenticar Implementado

2.7.2 AcrescentarDespesas

Morador fornece o tipo e o valor da despesa ao GUI. A mensagem enviada do GUI ao SGD, do SGD o Morador e do Morador para a Despesa são a mesma: uma mensagem a dizer que o pretendido é consultar a despesa. Despesa acrescenta essa despesa e, no fim, o GUI envia uma mensagem de sucesso ao Morador.

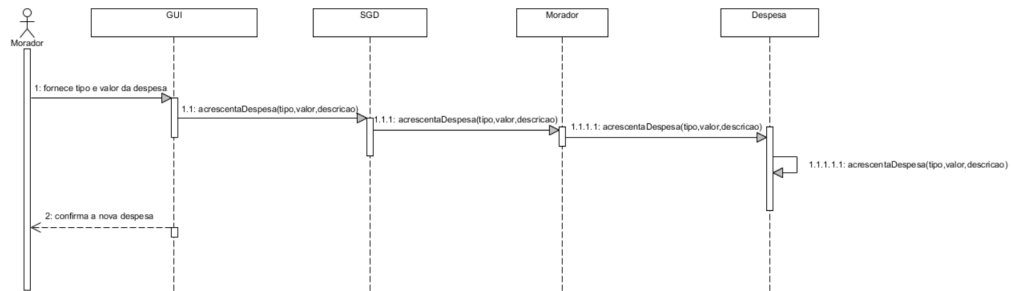


Figura 31: Diagrama de Sequência AcrescentarDespesas Implementado

2.7.3 ConsultarDespesas

Para consultar despesas, o Morador terá de fornecer o tipo das despesas que quer consultar ao GUI. Este irá invocar o método consultarDespesas que recebe um tipo de despesas e que terá de ser usado nas despesas de um morador e, por isso, é enviada a mensagem com o método do SGD para o Morador e por sua vez para a Despesa, visto que a consulta é feita sobre objetos do tipo Despesa. Não havendo despesas do tipo indicado, a interface informa o Morador de que não há nenhuma despesa desse tipo. Caso contrário, há despesas e o Morador poderá visualizar a lista das despesas desse tipo.

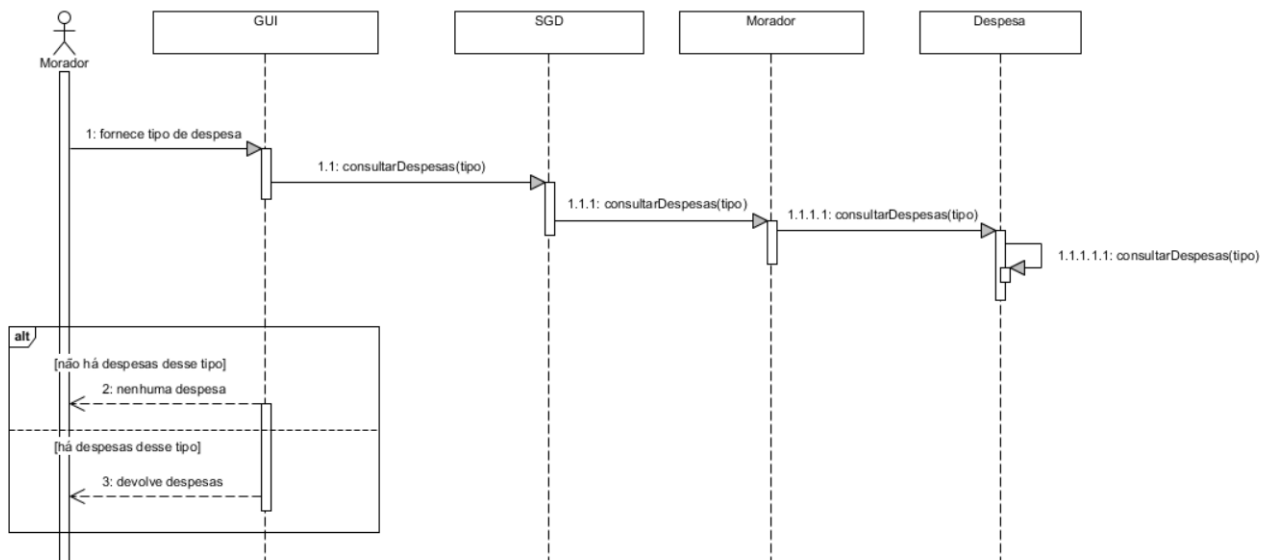


Figura 32: Diagrama de Sequência ConsultarDespesas(Morador) Implementado

2.7.4 PagarDespesa

A primeira troca de mensagem que ocorre quando um morador pretende pagar uma despesa é a mensagem trocada entre o Morador e o GUI que contém o id da despesa que se pretende pagar. Do GUI ao SGD e do SGD ao Morador circula uma mensagem que indica que é pretendido pagar a despesa. O Morador envia à Despesa um pedido para obter a despesa com o id passado pelo Morador e se esta despesa não existe, o GUI envia ao Morador uma mensagem de insucesso, indicando que não existe nenhuma despesa com o id dado. Se por outro lado o id existir, Despesa envia uma mensagem ao Pagamento a dizer que quer pagar a despesa com o tal id e o Pagamento paga essa despesa. No fim desta ação, o GUI confirma ao Morador que o pagamento foi efetuado.

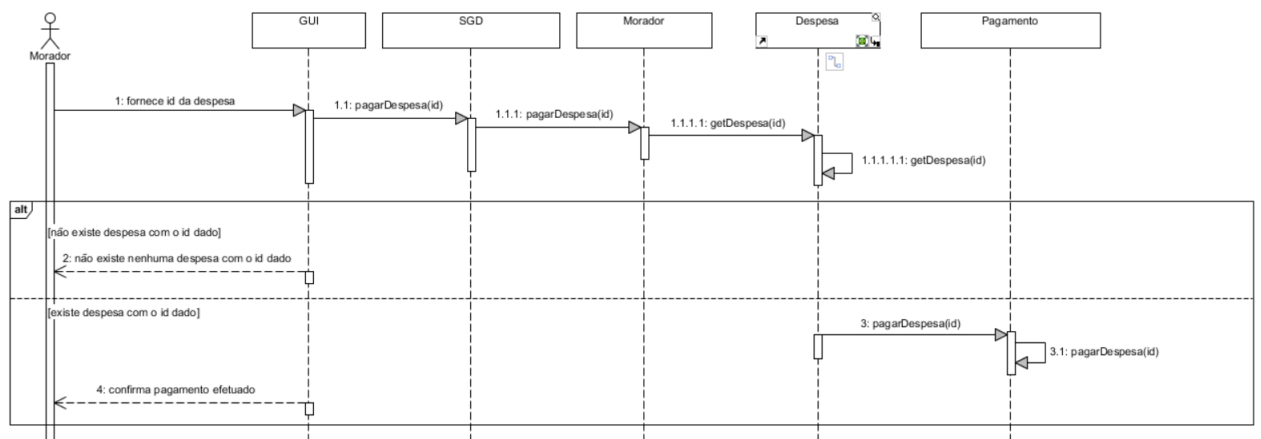


Figura 33: Diagrama de Sequência PagarDespesa Implementado

2.7.5 AcrescentarMorador

Neste diagrama, podemos ver que o primeiro passo para acrescentar um morador é o Senhorio indicar ao GUI o apartamento ao qual será acrescentado e o SGD envia ao Apartamento a mesma mensagem que é primeiramente enviada pelo GUI ao SGD: obter os detalhes do apartamento. O GUI apresenta ao Senhorio os dados do apartamento. De seguida, o Senhorio indica ao GUI os dados do morador e o GUI envia ao SGD, seguido do SGD para o Senhorio, seguido do Senhorio para o Morador, uma mensagem para acrescentar o morador. O Morador acrescenta esse morador e se o morador ainda não se encontrar no sistema, o GUI indica ao Senhorio que o morador foi acrescentado. Caso contrário, é trocada uma mensagem de insucesso.

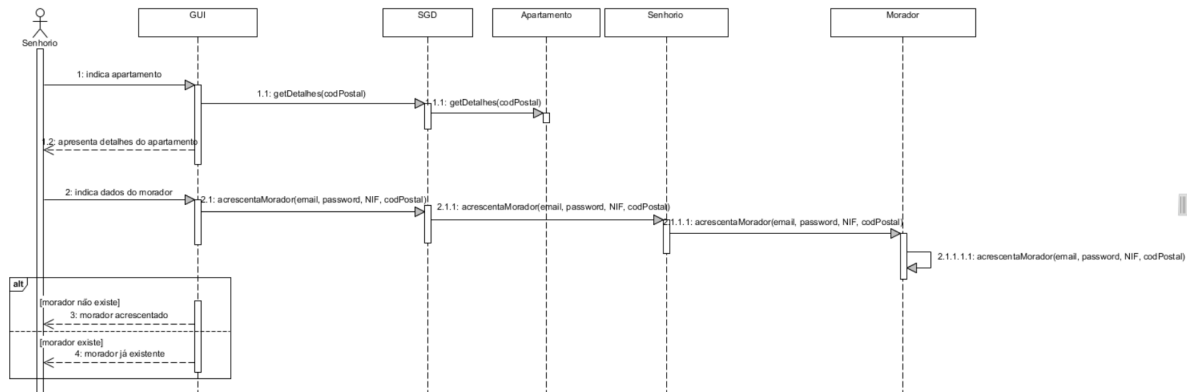


Figura 34: Diagrama de Sequência AcrescentarMorador Implementado

2.8 Diagrama de Package

No Diagrama de Package, temos três principais packages Camada de Interface, Camada de Negócio e Camada de Dados. O package da Camada de Negócio é o dono dos packages Despesas, Apartamentos, *facade* SGD, Pagamentos e Utilizadores.

O package Camada de Interface acede aos elementos exportados pelo package *facade* SGD e este package *facade* SGD acede aos elementos exportados pelo package Camada de Dados.

Qualquer alteração feita no package *facade* SGD afeta os package destino Despesa, Apartamentos, Pagamentos e Utilizadores.

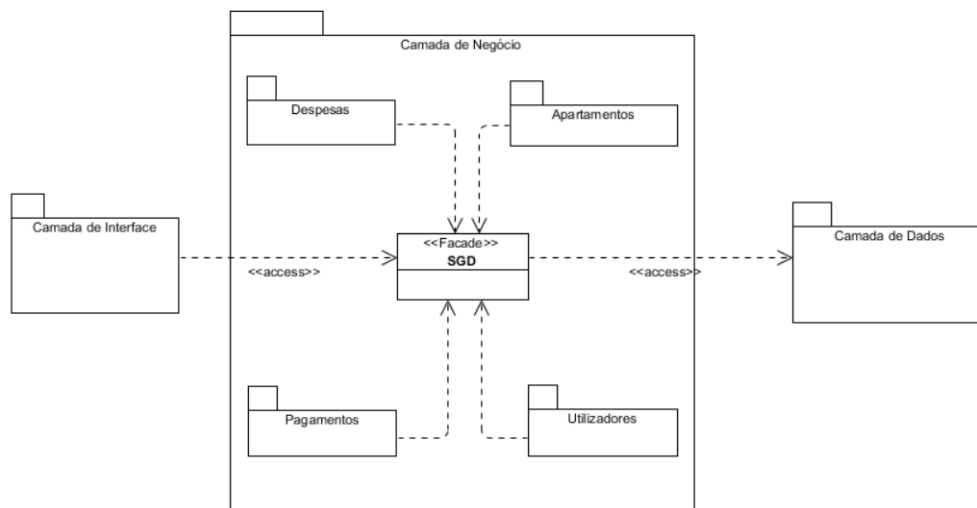


Figura 35: Diagrama de Package

2.9 Diagrama de Classe

Partindo dos relacionamentos do Modelo de Domínio e das suas multiplicidades, foi feito o Diagrama de Classe. Neste diagrama, apenas se encontram definidas as variáveis de cada classe. Neste diagrama, codB é chave na relação entre a classe SGD e a classe Senhorio, codA é chave na relação entre SGD e Apartamento, codM é chave na relação entre SGD e Morador e codD é chave na relação entre SGD e Despesa.

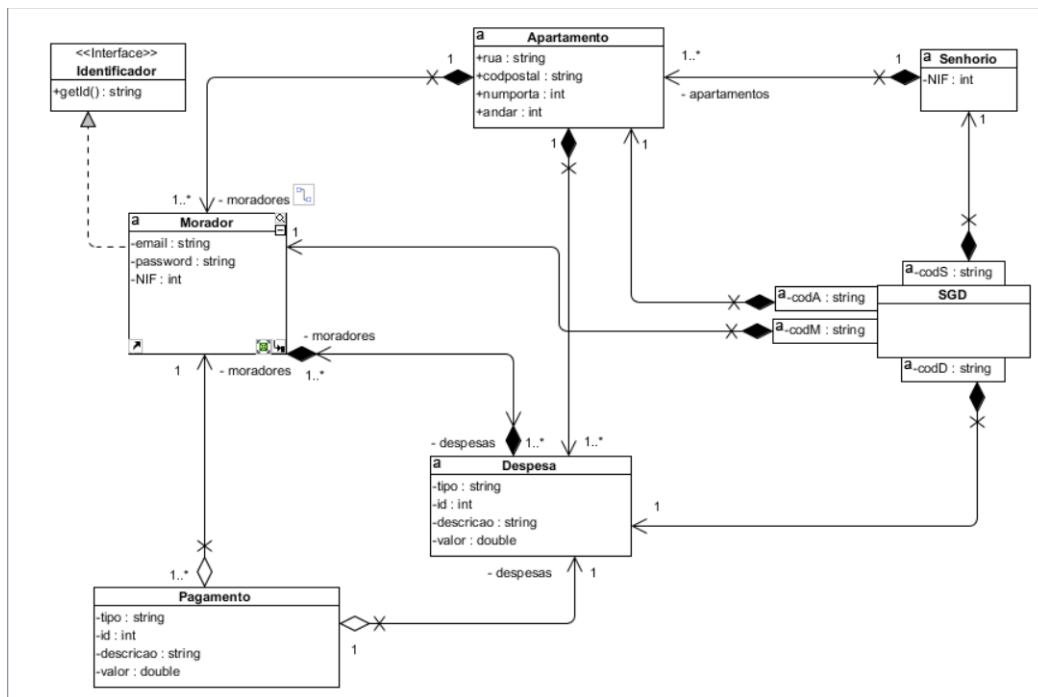


Figura 36: Diagrama de Classe

2.10 Diagrama de Classe com DAO's

A partir deste Diagrama de Casse começamos a implementar a persistência, tendo escolhido a visão onde a lógica está na base de dados, ou seja, usando DAO's.

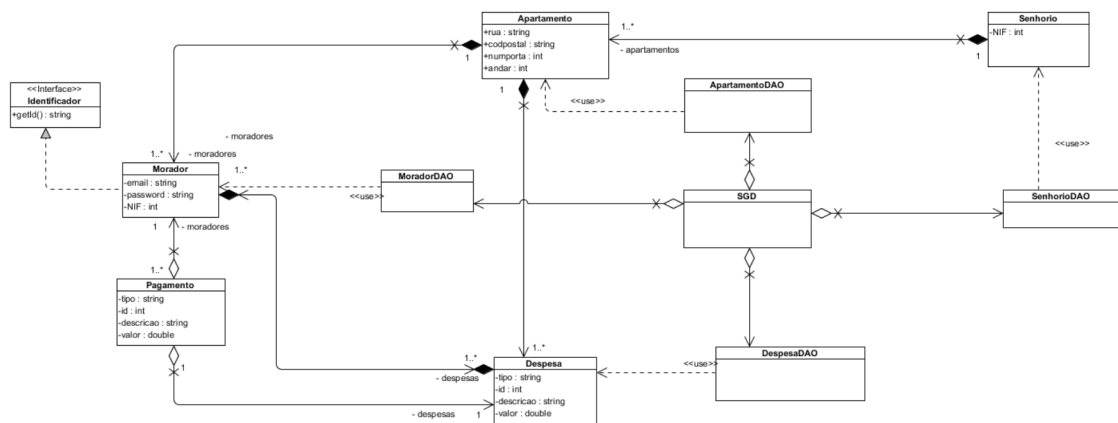


Figura 37: Diagrama de Classe com DAO's

2.11 Diagrama de Classe com Métodos

Este Diagrama de Classe encontra-se no nível de implementação, pois temos a definição concreta dos métodos a implementar, métodos esses usados para gerar código. Já estão definidas as API's de cada classe.

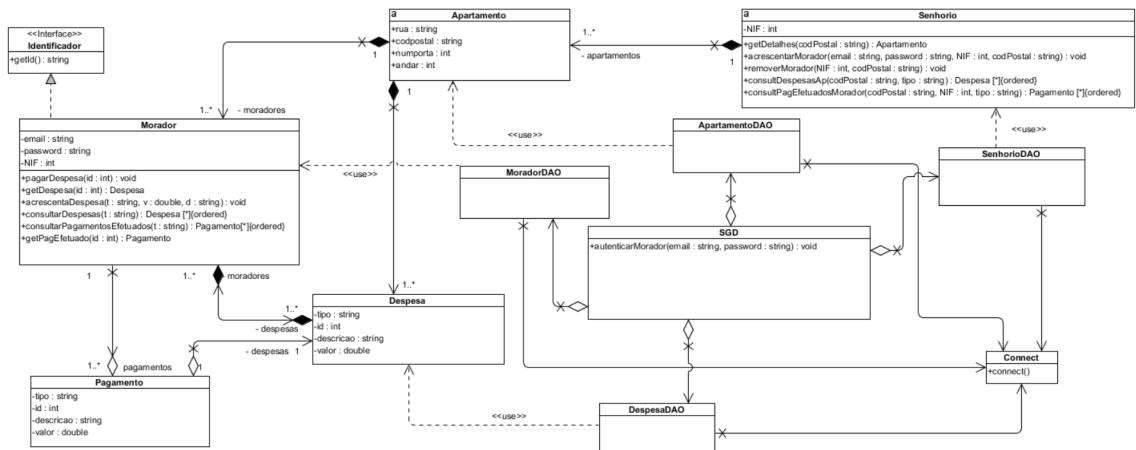


Figura 38: Diagrama de Classe com Métodos

2.12 Diagrama de Instalação

Neste diagrama de instalação, é possível ver os requisitos necessários a nível de hardware e software para que o modelo do nosso sistema possa ser instalado e fique funcional. Tanto o Morador como o Senhorio necessitam de um aparelho com um ambiente de execução adequado. Esse ambiente de execução irá estar ligado através do protocolo de rede TCP/IP ao ambiente de execução do servidor desta aplicação. O ambiente de execução do servidor terá ainda acesso a um componente, através de uma ligação JDBC, que será a base de dados em MySQL implementada neste servidor onde irão constar todos os dados necessários à aplicação.

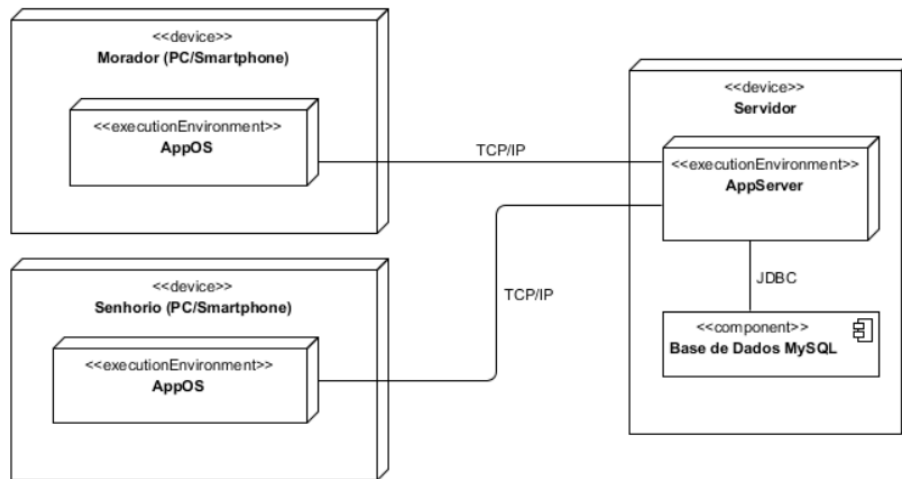


Figura 39: Diagrama de Instalação

3 Conclusão

Com o trabalho concluído, apercebemo-nos de que a parte elaborada até ao início da segunda parte do trabalho sofreu algumas alterações e acréscimos. Por exemplo, acrescentamos as modalidades de pagamento que estavam em falta na primeira fase.

Em relação aos Use Case, optamos por ter um ator Senhorio que funcionasse um pouco como um possível administrador, acima de tudo, por causa da gestão de quem está a habitar um dado apartamento num certo momento. O facto de poder consultar as despesas e os pagamentos já efetuados permite-lhe ter a noção do que se está a passar no apartamento, mesmo que algumas das despesas não necessitem totalmente de lhe ser comunicadas. Por isso, foi necessário dividir os atos de consultar despesas e pagar em dois Use Case visto que num deles apenas tem acesso o Morador, mas no outro têm acesso ambos os atores. As despesas já pagas num Use Case à parte estão destinadas a fornecer uma forma rápida ao Senhorio de saber quem já pagou e o que pagou. São também úteis para os moradores para que estes possam verificar rapidamente o que já pagaram. Apesar disso, julgamos que seria também possível num só Use Case consultar e pagar as despesas, mas optamos por manter a abordagem inicial.

Tivemos algumas dificuldades na realização dos diagramas de sequência na fase de implementação e também na realização do diagrama de classe. Além disso, tivemos também dificuldades na construção do código necessário, tendo ficado a realização da interface no *NetBeans* inacabada. Também não conseguimos, entre outras coisas, conectar uma base de dados.