



Universidade do Minho
Escola de Engenharia
Mestrado Integrado em Engenharia Informática

Unidade Curricular de Laboratórios de Informática IV

Ano Letivo de 2017/2018

Kiddo – Empresa de babysitting 24/7

Ana Paula Carvalho (A61855)

Guilherme Guerreiro (A73860)

João Barreira (A73831)

José Bastos (A74696)

Abril, 2018

Data de Recepção	
Responsável	
Avaliação	
Observações	

Kiddo – Empresa de babysitting 24/7

Ana Paula Carvalho (A61855)

Guilherme Guerreiro (A73860)

João Barreira (A73831)

José Bastos (A74696)

Fevereiro, 2018

Resumo

Este projeto é referente à Unidade Curricular de Laboratórios de Informática IV, do terceiro ano do Mestrado Integrado em Engenharia Informática da Universidade do Minho, que tem como principal objetivo a realização de um projeto de *software* robusto, composto por três etapas distintas, de acordo com a metodologia do *Rational Unified Process* (RUP).

A primeira destas três etapas (Fundamentação) diz respeito à identificação e caracterização geral da aplicação que vai ser desenvolvida. É nesta fase que é feita a fundamentação do seu desenvolvimento e a justificação em termos de modelo de negócio, bem como uma análise da viabilidade do sistema e respetivo plano de desenvolvimento.

A segunda etapa (Especificação) corresponde à descrição detalhada do *software* a desenvolver, utilizando a linguagem de modelação *Unified Modeling Language* (UML) para construir diversos diagramas como o Diagrama de *Use Cases*, Diagramas de Sequência, Diagramas de Classes, bem como a respetiva documentação.

A terceira etapa (Implementação) diz respeito à execução prática do *software*, tendo por base o planeamento efetuado nas fases anteriores. Corresponde, assim, à apresentação e caracterização geral da arquitetura do sistema e seus respetivos módulos. O trabalho é distribuído e é seguido o plano desenvolvimento segundo o que foi delineado na primeira fase. As ferramentas previamente acordadas são utilizadas para codificar a aplicação, respeitando os diagramas da segunda fase. A validação da aplicação criada será também incluída nesta última fase.

Área de Aplicação: Desenvolvimento de aplicação *web-based* e *mobile*, utilizando metodologias da Engenharia de *Software*.

Palavras-Chave: Engenharia de *Software*, Aplicações *mobile*, Aplicações *web-based*, Bases de Dados Relacionais, Engenharia de Requisitos, Modelo em cascata, *Rational Unified Process*, Metodologias de *Software*, Diagrama de *Gantt*, *Microsoft Project*, *Microsoft .NET*, *Microsoft SQL Server*.

Índice

Resumo	i
Índice	ii
Índice de Figuras	iv
1. Introdução	1
1.1. Contextualização	1
1.2. Apresentação do Caso de Estudo	1
1.3. Motivação e Objetivos	2
1.4. Justificação e Utilidade do Sistema	3
1.5. Definição da Identidade do Sistema	3
1.6. Identificação dos Recursos Necessários	4
1.7. Modelo do Sistema a Implementar	5
1.8. Definição de Medidas de Sucesso	6
1.9. Plano de Desenvolvimento	7
2. Requisitos	8
2.1. Requisitos do Utilizador	8
2.2. Requisitos do Sistema	10
2.2.1. Funcionais	10
2.2.2. Não Funcionais	11
3. Modelos do Sistema	12
3.1. Diagramas	12
3.1.1 Diagramas de Use Case	12
3.1.2 Especificação de Use Cases	13
3.1.3 Diagramas de Sequência	18
3.1.4 Diagramas de Sequência de Subsistemas	27
3.1.5 Diagramas de Atividade	35
3.1.6 Modelo de Domínio	41
3.1.7 Diagrama de Classes	42
3.1.8 Máquinas de Estado	42
4. Base de Dados	44
4.1. Modelo Conceptual	44
4.1.1 Diagrama do Modelo Conceptual	44
4.1.2 Identificação de Identidades	44

4.1.3 Identificação dos Relacionamentos	45
4.1.4 Identificação e Associação de Atributos	45
4.1.5 Determinação do Domínio dos Atributos	46
4.1.6 Identificação de Chaves Primárias	48
4.2. Modelo Lógico	48
4.2.1 Diagrama do Modelo Lógico	48
4.2.2 Derivação de Relacionamentos	49
4.2.3 Validação Através da Validação	49
4.2.4 Viabilidade de Crescimento	49
5. <i>Mockups</i> da Interface	51
5.1. Login – mobile	52
5.2. Home – mobile	53
5.3. Lista de serviços – mobile	54
5.4. Serviço - mobile	55
5.5. Aceitação de serviços – mobile	56
5.6. Login – web	57
5.7. Home – web	58
5.8. Requisitar serviço – web	59
5.9. Contratar – web	60
6. Conclusões e Trabalho Futuro	61
7. Referências	62

Índice de Figuras

Figura 1 - Maquete do sistema	5
Figura 2 - Diagrama de <i>Gantt</i>	7
Figura 3 - Diagrama de Use Cases	12
Figura 4 - Especificação de Use Case "Aceitar pedido de serviço"	13
Figura 5 - Especificação de Use Case "Adicionar à lista de favoritos"	14
Figura 6 - Especificação de Use Case "Aprovar candidatura"	14
Figura 7 - Especificação de Use Case "Consultar catálogo de prestadores"	14
Figura 8 - Especificação de Use Case "Consultar lista de candidaturas"	15
Figura 9 - Especificação de Use Case "Consultar lista de pedidos de serviço"	15
Figura 10 - Especificação de Use Case "Solicitar serviço"	16
Figura 11 - Especificação de Use Case "Submeter candidatura"	16
Figura 12 - Especificação de Use Case "Submeter pontuação"	17
Figura 13 - DS de "Aceitar/recusar candidatura a Babysitter"	18
Figura 14 - DS de "Aceitar/recusar pedido de serviço"	19
Figura 15 - DS de "Adicionar Babysitter à lista de favoritos"	20
Figura 16 - DS de "Consultar lista de Babysitters"	21
Figura 17 - DS de "Consultar lista de candidaturas a Babysitter"	22
Figura 18 - DS de "Consultar lista de pedidos de serviço"	23
Figura 19 - DS de "Enviar pedido de serviço"	24
Figura 20 - DS de "Remover Babysitter do sistema"	25
Figura 21 - DS de "Submeter candidatura"	26
Figura 22 – DSS de "Aceitar/recusar candidatura a Babysitter"	27
Figura 23 - DSS de "Aceitar/recusar pedido de serviço"	28
Figura 24 - DSS de "Adicionar Babysitter à lista de favoritos"	29
Figura 25 - DSS de "Consultar lista de Babysitters"	30
Figura 26 - DSS de "Consultar lista de candidaturas a Babysitter"	31
Figura 27 - DSS de "Consultar lista de pedidos de serviço"	31
Figura 28 - DSS de "Enviar pedido de serviço"	32
Figura 29 - DSS de "Remover Babysitter do sistema"	33
Figura 30 - DSS de "Submeter candidatura"	34
Figura 31 - DA de "Aceita pedido de serviço"	35

Figura 32 - DA de "Adiciona a lista de favoritos"	36
Figura 33 - DA de "Aprova babysitter"	37
Figura 34 - DA de "Consulta catálogo de prestadores"	37
Figura 35 - DA de "Consulta lista de candidaturas"	38
Figura 36 - DA de "Consulta lista de pedidos de serviço"	38
Figura 37 - DA de "Retira babysitter do sistema"	39
Figura 38 - DA de "Solicita serviço"	39
Figura 39 - DA de "Submete candidatura a babysitter"	40
Figura 40 - DA de "Submete pontuação"	40
Figura 41 - Modelo de Domínio	41
Figura 42 - Diagrama de Classes	42
Figura 43 - Máquina de Estado (website)	43
Figura 44 - Máquina de Estado (app)	43
Figura 45 - Modelo Conceptual da Base de Dados	44
Figura 46 - Modelo Lógico da Base de Dados	48
Figura 47 - <i>Mockup</i> do ecrã de login (aplicação móvel)	52
Figura 48 - <i>Mockup</i> do ecrã de home (aplicação móvel)	53
Figura 49 - <i>Mockup</i> da lista de serviços (aplicação móvel)	54
Figura 50 - <i>Mockup</i> do ecrã de serviço (aplicação móvel)	55
Figura 51 - <i>Mockup</i> de aceitação de serviços (aplicação móvel)	56
Figura 52 - <i>Mockup</i> do ecrã de login (web)	57
Figura 53 - <i>Mockup</i> do ecrã de home (web)	58
Figura 54 - <i>Mockup</i> do ecrã de requisitar serviço (web)	59
Figura 55 - <i>Mockup</i> do ecrã de fazer pedido de serviço (web)	60

1. Introdução

1.1. Contextualização

Nos dias que correm, as pessoas utilizam uma grande variedade de serviços que estão disponíveis de forma rápida nos seus dispositivos mais próximos, como *smartphones* ou computadores pessoais. Vivemos num mundo totalmente conectado, em que existe uma grande facilidade em, por exemplo, fazer compras sem sair de casa ou consultar o saldo bancário sem sair do escritório.

Apesar de todo este comodismo, o mundo atual é mais do que nunca frenético e atribulado, tendo passado a haver uma simbiose perigosa entre a vida pessoal e profissional – anteriormente desconectadas –, o que faz com que os trabalhadores dos dias de hoje estejam muitas mais horas ligados ao trabalho do que as estipuladas no seu contrato. Passou, por isso, a haver um maior lugar para os imprevistos e para a supressão da vida familiar em favor do trabalho.

Um destes casos, é a vida dos pais que se encontra ainda mais preenchida, o que faz com que, por vezes, não possam cuidar permanentemente dos seus filhos. Assim sendo, surge a necessidade de contratar serviços especializados que possam colmatar esta falha, permitindo aos pais uma vida um pouco mais cómoda e organizada.

Neste contexto, a *Kiddo* pretende afirmar-se como uma solução e uma ajuda na vida destes mesmos pais, oferecendo um serviço simples e ao dispor de qualquer pessoa.

1.2. Apresentação do Caso de Estudo

O projeto a ser desenvolvido deverá ser dividido em duas interfaces de utilização, uma para utilizadores e outra para funcionários da empresa. Para os utilizadores, será criada uma plataforma *web* de modo a lhes ser permitido:

- Encontrar amas disponíveis ordenadas por distância/nota de avaliação;
- Encontrar amas com determinadas características;
- Pedir um serviço;

- Manter e adicionar a uma lista de amas favoritas;
- Avaliar amas;
- Criar preferências e notas relevantes.

Graças a estas funcionalidades, o utilizador conseguirá encontrar uma ama que se adeque às suas exigências e que garanta a sua satisfação.

Para os prestadores do serviço, será então disponibilizada uma aplicação móvel que notifica aquando o surgimento de um pedido por parte de um cliente e informa a morada do mesmo, número de crianças e notas adicionais relevantes.

Os *babysitters* vinculados ao *Kiddo* e todas as suas características serão obtidas a partir de uma base de dados online regida por proprietários do sistema, sendo estes capazes de:

- Adicionar/retirar amas;
- Associar características a uma ama;
- Estipular um raio de atividade de uma ama.

Estas funções não estarão incluídas na aplicação móvel, estando reservadas a uma página *online* exclusiva para o propósito.

1.3. Motivação e Objetivos

O motivo por detrás da conceção deste projeto, prende-se pelo facto de não existir um mercado desenvolvido e especializado no ramo da prestação de serviços de *babysitting*. Assim sendo, a *Kiddo* pretende marcar uma posição de protagonismo, distinguindo-se pela criação de uma plataforma inovadora que disponibilize um serviço simples, cómodo e rápido para os pais ativos dos dias de hoje.

Desta forma, a *Kiddo* tem como objetivo oferecer uma opção confiável, pondo ao dispor de qualquer pai ou mãe um amplo leque de *babysitters* especializados e aprovados previamente. Queremos, então, que com poucos cliques (ou toques) os pais possam ver as suas vidas facilitadas e, ao mesmo, oferecer um maior público-alvo aos cuidadores de crianças e jovens.

A *Kiddo* surge por forma a melhorar a vida de um grande número de pais e respetivas famílias, ambicionado uma larga adesão que irá contribuir para a afirmação deste tipo de serviços na sociedade atual.

1.4. Justificação e Utilidade do Sistema

A ideia para o desenvolvimento desta aplicação surgiu através do facto de a equipa ter constatado que arranjar um *babysitter* nem sempre se revela uma tarefa fácil.

Os pais dos dias de hoje, mais do que nunca, vivem uma vida atarefada e frenética, que, muitas vezes, impossibilita uma presença permanente junto dos seus filhos. Atualmente, os pais recorrem frequentemente aos avós e outros familiares das crianças de forma a suprir esta necessidade. No entanto, esta nem sempre é a opção mais cómoda para os pais ativos e deslocalizados da família. Assim sendo, nesse caso, as alternativas são escassas ou pura e simplesmente inexistentes.

É aqui que a *Kiddo* pretende atuar, preenchendo uma lacuna de mercado no que toca a aplicações móveis e *web-based* de requisição de serviços de *babysitting*. Através de uma aplicação simples, a *Kiddo* tem como missão revolucionar este panorama, tanto do lado dos pais que pretendem uma solução cómoda e rápida que se adapte ao seu estilo de vida, como dos cuidadores que pretendem atingir um maior público-alvo.

Tendo em conta este estudo de mercado, a equipa considerou viável este projeto.

1.5. Definição da Identidade do Sistema

Foi-nos proposto criar uma aplicação que funcionasse como um sistema de prestação de serviços ao domicílio, capaz de funcionar 24 horas por dia, 7 dias por semana. Surgiu, então, a aplicação *Kiddo* - um serviço de *babysitting*.

Descrevendo mais detalhadamente o nosso sistema, existirão dois tipos de utilizadores: os requisitadores do serviço e os *babysitters*.

O requisitador terá acesso a um catálogo para consulta dos *babysitters* disponíveis, onde pode pesquisar segundo alguns critérios de localização ou experiência, por exemplo. De seguida requisita o serviço, estando sujeito à disponibilidade dos prestadores. Se não acontecer nenhum entrave, ambas as entidades acordam as condições do serviço e procedem ao agendamento. Finalmente, o serviço é realizado e é efetuada a cobrança, com envio via email ou SMS da fatura, tendo o utilizador a opção de avaliar o serviço atribuindo uma avaliação.

Já quem providencia o serviço utiliza o sistema de outra forma, efetuando o registo e sujeitando-se a aprovação. Após ser inserido no sistema, recebe o contacto dos possíveis clientes, procedendo-se de igual forma. Através da aplicação móvel, os *babysitters* poderão, por exemplo, visualizar os agendamentos e respetivas localizações.

1.6. Identificação dos Recursos Necessários

Ao nível do desenvolvimento do *website*, vamos utilizar o *Bootstrap* para nos auxiliar no processo de codificação do HTML, CSS e *Javascript* para a aplicação *web-based*. Esta ferramenta é bastante popular no panorama atual do desenvolvimento *web*. Possui alguns componentes pré-fabricados como barras de navegação, caixas de diálogo, formulários, botões, etc. cujas propriedades podem ser alteradas pelo grupo, por forma a criar um *website* moderno e responsivo às dimensões do navegador.

Além disso, iremos precisar de utilizar um serviço de localização GPS na nossa aplicação móvel, por forma a permitir aos *babysitters* saber como chegar a casa no cliente. Neste aspeto, o grupo utilizará o serviço *Google Maps* por ter uma API completa, bem documentada e uma grande comunidade de utilizadores.

Para além destes aspetos, o grupo irá utilizar as seguintes ferramentas no decorrer do desenvolvimento do projeto:

- *Microsoft Excel* será utilizado para o planeamento do desenvolvimento do projeto (uso de um *template* para diagramas de *Gantt*);
- *Visual Paradigm* para a modelação do projeto;
- *Microsoft Word* para a escrita dos relatórios/documentação;
- *Microsoft PowerPoint* para as apresentações;
- *Microsoft SQL Server* para a persistência dos dados;
- *Microsoft Visual Studio* irá ser o nosso *IDE*;
- *Microsoft .NET C#* para a codificação da aplicação.

1.7. Modelo do Sistema a Implementar

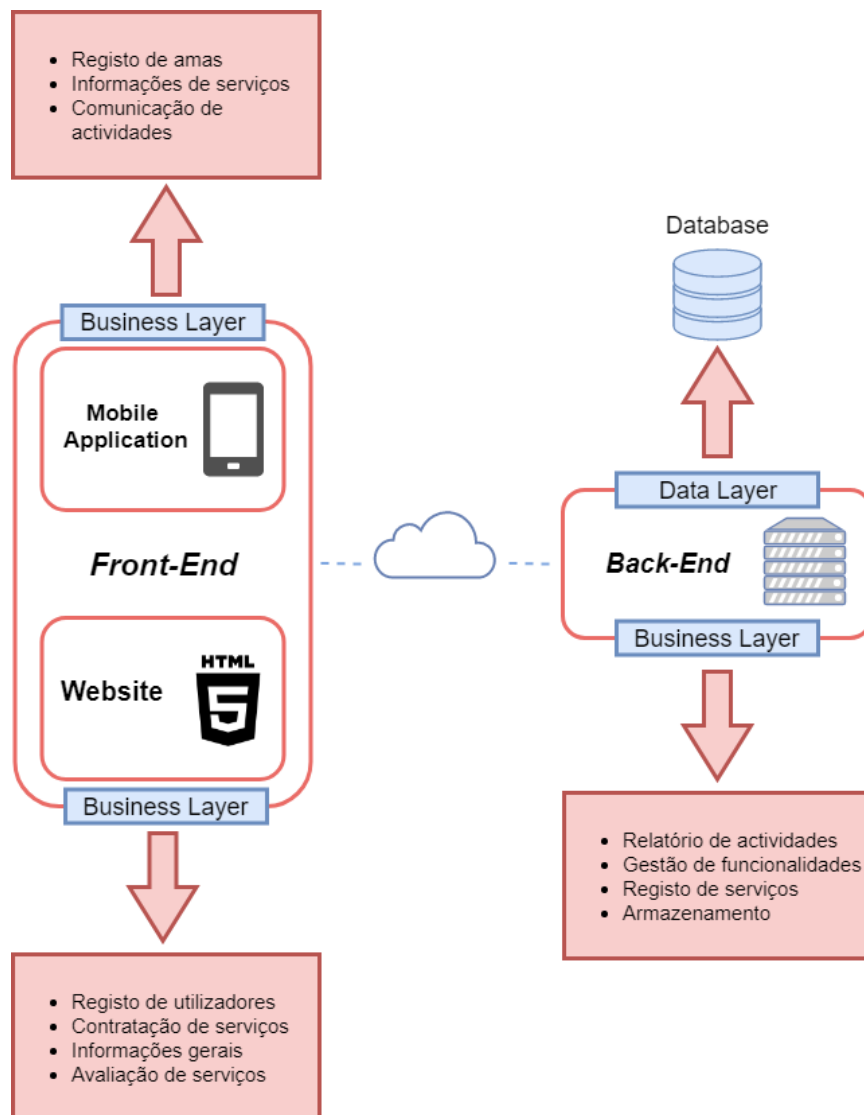


Figura 1 - Maquete do sistema

O nosso sistema é composto por dois principais componentes: *Front-End* e *Back-End*.

O *Front-End* tem duas vertentes: aplicação e website. A aplicação é usada pelos de serviço de forma a serem mais facilmente contactados e de obterem com mais rapidez as informações do serviço contratado pelo utilizador. Poderão, de uma forma simples, verificar os seus agendamentos futuros, bem como, por exemplo, obter as localizações respetivas.

O *website* irá servir de suporte para o utilizador, dando acesso à contratação de serviços, informações sobre o funcionamento do sistema e avaliações.

Este *website* será o principal portal do serviço do ponto de vista dos clientes, sendo que a aplicação móvel será uma ferramenta auxiliar dos *babysitters*.

Já o *Back-End* corresponde ao servidor que sustenta a nossa aplicação, bem como à implementação das funcionalidades escondidas aos utilizadores e *babysitters* como, por exemplo, o registo de novos utilizadores no sistema. Estando este sempre em constante atividade, o servidor será o local onde ocorrerá o armazenamento de todos os dados dos utilizadores e dos serviços e a gestão das funcionalidades internas do sistema.

É, então, importante que exista uma correta ligação entre o *Front-End* e o *Back-End*, por forma a apresentar os dados corretos aos clientes e *babysitters* de uma forma rápida e confiável.

1.8. Definição de Medidas de Sucesso

Por forma a avaliar o êxito do produto final deste projeto, foi definido um conjunto de medidas de sucesso.

Os proprietários do sistema pretendem que a aplicação renda, ao fim de um ano, no mínimo 20% do investimento inicial para a sua produção e conseguir recrutar cerca de 15 amas, de modo a assegurar um crescimento do serviço nos anos subsequentes.

1.9. Plano de Desenvolvimento

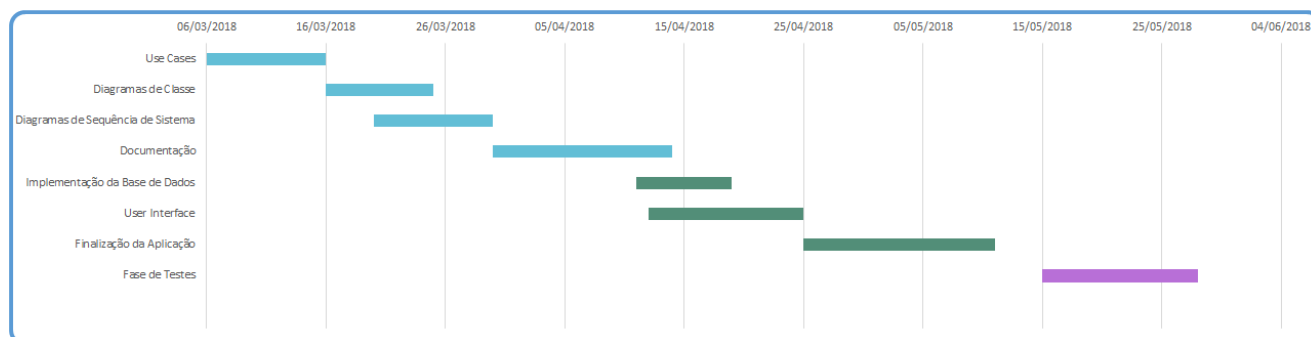


Figura 2 - Diagrama de *Gantt*

O arranque deste projeto consistiu, primeiramente, pela fundamentação do tema escolhido. Foi feita um estudo de mercado do ramo e uma pesquisa esmerada para recolher toda a informação relevante ao serviço de *babysitting*, de modo a inteirar o grupo das necessidades dos possíveis utilizadores da aplicação. Deste modo, conseguimos perceber claramente como o *Kiddo* poderia facilitar o dia-a-dia dos seus utilizadores.

Feita a recolha de informação, passou-se então a contextualizar o problema e apresentar do caso de estudo, motivação e objetivos. Em paralelo foi feito uma maquete do sistema, idealizando a arquitetura da aplicação. A partir deste ponto, foi feito o planeamento das restantes fases de desenvolvimento, tendo-se definido um conjunto de medidas de sucesso por forma a avaliar o trabalho futuro e produto final resultante.

2. Requisitos

O levantamento de requisitos é uma fase de extrema importância para garantir o bom desenvolvimento do software, garantindo um entendimento sólido do que é esperado em termos de funcionalidades quando alcançado o produto final. Serão representados, de seguida, requisitos do utilizador e do sistema, recolhidos minuciosamente através de uma entrevista com o cliente e analisados cuidadosamente.

Nos requisitos do sistema será também feita a distinção entre requisitos funcionais e não funcionais.

2.1. Requisitos do Utilizador

No sistema da *Kiddo* inserem-se três tipos entidades, distintas nas suas funcionalidades - *babysitter*, cliente e administrador. Assim sendo, cada uma delas terá necessidades diferentes que passaremos a enumerar:

1. Cada cliente deve poder aceder aos perfis de todos os prestadores vinculados ao sistema.
2. Cada serviço é solicitado por um cliente a um prestador de serviço. No pedido do serviço são explicitadas as particularidades deste: endereço do serviço, número de crianças, idades, hora, duração e informações adicionais que o cliente considere de relevância.
3. Após efetuado o pedido de serviço, é esperado que o prestador tenha a escolha de o realizar ou não, mediante os detalhes deste.
4. Os prestadores do serviço devem ter um perfil público, acessível a qualquer cliente, que contenha o seu nome, idade, experiência na área, uma pequena biografia, fotografia, avaliação dos clientes e número de serviços realizados na *Kiddo*.
5. Os clientes devem igualmente ter um perfil, acessível ao prestador solicitado, com nome, localidade e fotografia.

6. Os prestadores devem conter, oculto do público, o número de conta bancária de modo a que a sua comissão possa ser depositada após a realização do serviço.
7. Também os clientes devem ter ocultos os dados necessários para ser efetuado os pagamentos dos serviços solicitados.
8. Cada prestador deve ser sujeito a uma análise cuidada antes de ser inserido no sistema – deve submeter uma candidatura, com o seu email, os seus dados, currículo, registo criminal. Pretende-se que esta candidatura seja revista por um administrador após a sua inserção.
9. Não se pretende que os prestadores possam editar informações no seu perfil relativas ao seu progresso profissional – qualquer alteração no seu currículo deve ser reportada ao administrador, sendo adicionada ao perfil por este.
10. Para cada serviço realizado - e após o seu pagamento – deve ser gerada uma fatura e enviada ao cliente.
11. Pretende-se que a aplicação facilite a comunicação entre cliente e prestador enquanto o serviço está a decorrer.

2.2. Requisitos do Sistema

2.2.1. Funcionais

- 1.1. Será facultado ao cliente um catálogo de modo a consultar os perfis dos prestadores de serviço e conseguir interagir com estes.
- 2.1. Para que seja criado um pedido de serviço é necessário que o cliente tenha um registo e sessão iniciada. Garantindo isto, é gerado um pedido de serviço que fica pendente até aceitação ou recusa por parte do prestador escolhido.
- 3.1. O pedido de serviço, depois de gerado, é associado a um prestador que é notificado.
- 3.2. Os pedidos recebidos são listados quando o prestador os quiser consultar e escolhe se os quer realizar ou não.
- 3.3. Quando um pedido é aceite é gerado um serviço, com contactos e as informações contidas no pedido de serviço. O prestador pode aceder aos seus serviços ativos na sua conta.
- 4.1. Cada prestador terá informações públicas, contidas num perfil, que poderão ser acedidas pelos requisitadores através do catálogo.
- 5.1. Os requisitadores também possuirão dados públicos contidos num perfil que poderá ser acedido pelos requisitadores através do pedido de serviço.
- 6.1. Além dos dados públicos do perfil, os prestadores terão também associado um número de conta, que será mantido privado e de acesso restrito os administradores.
- 7.1. Da mesma forma, os dados de pagamentos do requisitador serão mantidos privados, acessíveis apenas pela administração.
- 8.1. Será criado um método de submissão de candidaturas, que serão acedidas apenas pela administração.
- 9.1. Quando uma candidatura é aprovada, o registo do prestador será efetuado pelo administrador com base nos dados enviados nesta e é criado o seu perfil. As informações relativas à experiência profissional só poderão ser editadas pelo administrador e o número de serviços e pontuação serão gerados pela aplicação, não podendo nunca ser editados.
- 10.1. Após um serviço ser dado como concluído, é gerada uma fatura a ser paga pelo requisitador.
- 11.1. Quando um prestador aceita um serviço são apresentados, a cada um dos intervenientes da ação, o contacto do outro de modo a que a comunicação seja facilitada.

2.2.2. Não Funcionais

Visto não termos necessidade de nenhum componente de *hardware* em específico, não existem requisitos não funcionais.

3. Modelos do Sistema

3.1. Diagramas

3.1.1 Diagramas de Use Case

Para uma melhor representação do funcionamento global do sistema foi desenvolvido um diagrama de Use Case. Este reflete a interação dos vários tipos de utilizadores com a aplicação.

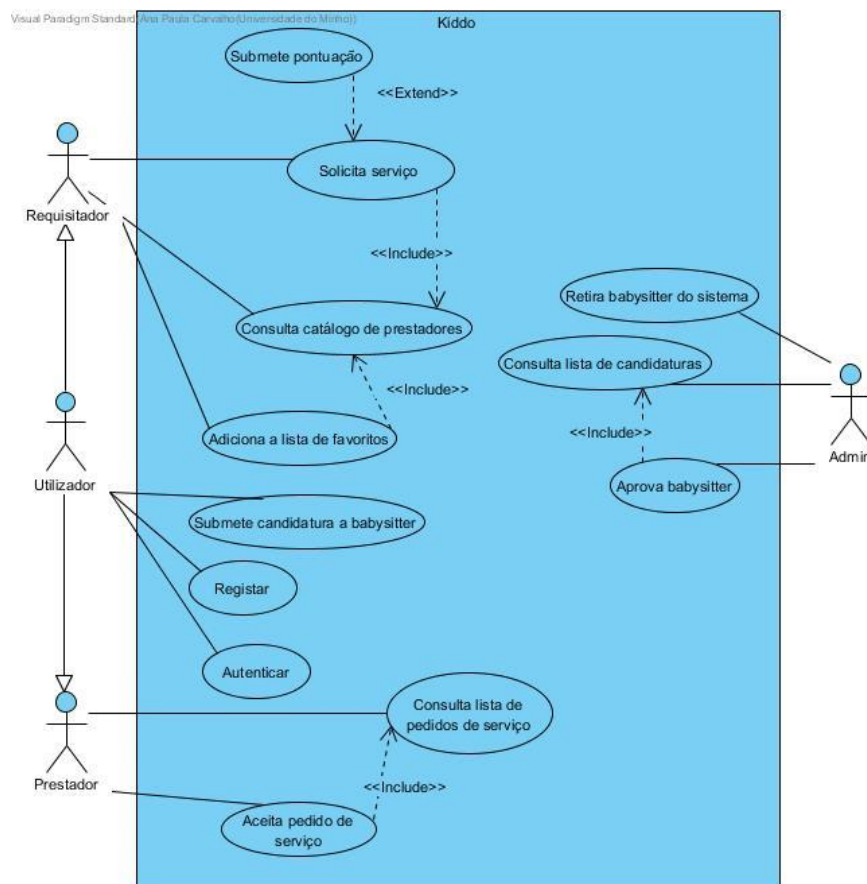


Figura 3 - Diagrama de Use Cases

No nosso caso foram definidos três tipos de utilizadores, cada um com acesso a funcionalidades diferentes, conforme o demonstrado na imagem. Em particular, a entidade utilizador, refere-se a um utilizador sem sessão iniciada que poderá fazer registo, autenticar-se ou ainda submeter uma proposta a fim de se associar à *Kiddo* como *babysitter*. No caso do prestador, este não precisará de efetuar qualquer tipo de registo visto que é o administrador do sistema que o realiza após a aprovação da candidatura.

3.1.2 Especificação de Use Cases

Para uma melhor compreensão dos objetivos dos *Use Cases* descritos no diagramas, foram elaboradas especificações dos mesmos. Estas especificações simulam os vários comportamentos do sistema para cada uma das interações com o utilizador, acabando por ajudar a perceber melhor como deverá ser a implementação da aplicação.

Incluímos, nesta secção, as diversas especificações desenvolvidas pelo grupo.

Use case: Aceitar pedido de serviço. Pré condição: É selecionado um pedido da lista de pedidos de serviço. Pós-condição: É enviada uma resposta do pedido de serviço.		
	Actor	Sistema
Comportamento Normal	1. Indica que pretende realizar o serviço.	2. Envia resposta ao pedido de serviço. 3. Mostra informação de contacto. 4. Sucesso.
Alternativa (passo 1)	1.1. Indica que pretende recusar o serviço.	1.2. Envia resposta ao pedido de serviço. 1.3. Cancela pedido de serviço.

Figura 4 - Especificação de Use Case "Aceitar pedido de serviço"

Use case: Adicionar à lista de favoritos. Pré condição: É selecionada um prestador do catálogo. Pós-condição: É adicionado um prestador à lista de favoritos.		
	Actor	Sistema
Comportamento Normal	1. Indica que quer adicionar prestador à lista de favoritos.	2. Verifica se prestador já se encontra na lista. 3. Acrescenta à lista de favoritos. 4. Sucesso.
Exceção (passo 3)		3.1. Reporta que o prestador já se encontra na lista de favoritos.

Figura 5 - Especificação de Use Case "Adicionar à lista de favoritos"

Use case: Aprovar candidatura. Pré condição: É selecionado uma candidatura da lista de candidaturas. Pós-condição: É enviada uma resposta a uma candidatura.		
	Actor	Sistema
Comportamento Normal	1. Indica que pretende analisar candidatura. 3. Aprova a candidatura.	2. Retorna os dados da candidatura. 4. Regista o email como prestador. 5. Gera código de acesso. 4. Envia a resposta e código ao candidato.
Exceção (passo 3)	3.1. Reprova a candidatura.	3.2. Envia resposta ao candidato. 3.3. Elimina candidatura.

Figura 6 - Especificação de Use Case "Aprovar candidatura"

Use case: Consultar catálogo de prestadores. Pré condição: Utilizador encontra-se autenticado. Pós-condição: -		
	Actor	Sistema
Comportamento Normal	1. Acede ao catálogo de prestadores.	2. Calcula lista de prestadores. 3. Apresenta lista ordenada por pontuação. 4. Sucesso.
Exceção (passo 2)		2.1. Reporta não existirem prestadores no sistema. 2.2. Aborta acesso ao catálogo.

Figura 7 - Especificação de Use Case "Consultar catálogo de prestadores"

Use case: Consultar lista de candidaturas. Pré condição: O Admin encontra-se autenticado. Pós-condição: -		
	Actor	Sistema
Comportamento Normal	1. Acede à lista de candidaturas.	2. Calcula lista de candidaturas. 3. Apresenta lista. 4. Sucesso.
Exceção (passo 2)		2.1. Reporta que não existem candidaturas. 2.2. Aborta acesso à lista de candidaturas.

Figura 8 - Especificação de Use Case "Consultar lista de candidaturas"

Use case: Consultar lista de pedidos de serviço. Pré condição: O prestador encontra-se autenticado. Pós-condição: -		
	Actor	Sistema
Comportamento Normal	1. Acede à lista de pedidos de serviço.	2. Calcula lista de pedidos de serviço. 3. Apresenta lista. 4. Sucesso.
Exceção (passo 2)		2.2. Reporta que não existem pedidos de serviço. 2.3. Aborta acesso à lista.

Figura 9 - Especificação de Use Case "Consultar lista de pedidos de serviço"

Use case: Solicitar serviço. Pré condição: É selecionada uma babysitter do catálogo. Pós-condição: É lançado um pedido de serviço.		
	Ator	Sistema
Comportamento Normal		1. Pede inserção de dados.
	2. Insere hora, duração, morada do serviço, número de crianças, idades e cuidados especiais.	
	3. Confirma pedido de serviço.	
		4. Apresenta custo do serviço.
	5. Confirma pagamento.	4. Envia pedido de serviço.
Exceção 1 (passo 5)	5.1. Indica não pagamento.	5.2. Aborta pedido de serviço.

Figura 10 - Especificação de Use Case "Solicitar serviço"

Use case: Submeter candidatura. Pré condição: - Pós-condição: É submetida uma candidatura.		
	Actor	Sistema
Comportamento Normal	1. Indica que quer submeter candidatura.	
	3. Insere nome, idade, currículo, registo criminal, email.	2. Solicita dados.
	4. Confirma registo.	
		5. Valida dados. 6. Admite candidatura. 7. Informa que a candidatura vai ser avaliada por Admin.
Exceção (passo 5)		5.1. Informa que o email inserido não é válido. 5.2. Aborta submissão de candidatura.

Figura 11 - Especificação de Use Case "Submeter candidatura"

Use case: Submeter pontuação Pré condição: É feito uma solicitação de serviço e este é efectuado. Pós-condição: É adicionada uma pontuação à prestadora do serviço.		
	Actor	Sistema
Comportamento Normal	2. Insere nota desejada	1. Pede inserção de pontuação
		3. Valida pontuação 4. Adiciona a pontuação ao perfil do prestador do serviço.
Exceção 1 (passo 3)		3.1. Reporta pontuação inválida 3.2. Aborta submissão de pontuação

Figura 12 - Especificação de Use Case "Submeter pontuação"

3.1.3 Diagramas de Sequência

De forma a complementar a informação descrita e detalhada sobre as funcionalidades fornecidas através dos diagramas de Use Cases, foram desenvolvidos diagramas de sequência.

De um modo mais generalista, será possível observar, através destes, todo o conjunto de operações que são relevantes para o bom funcionamento da nossa aplicação.

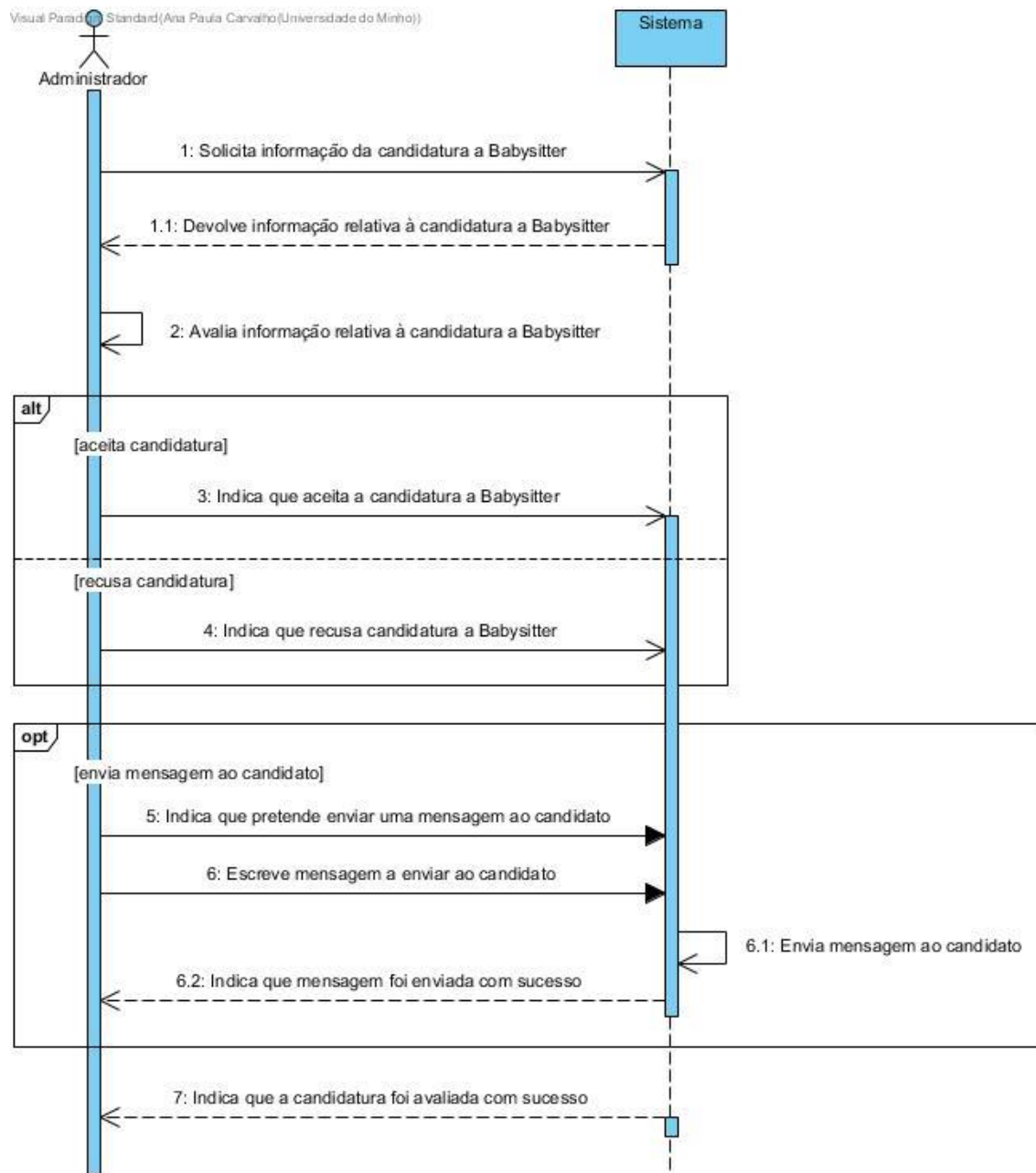


Figura 13 - DS de "Aceitar/recusar candidatura a Babysitter"

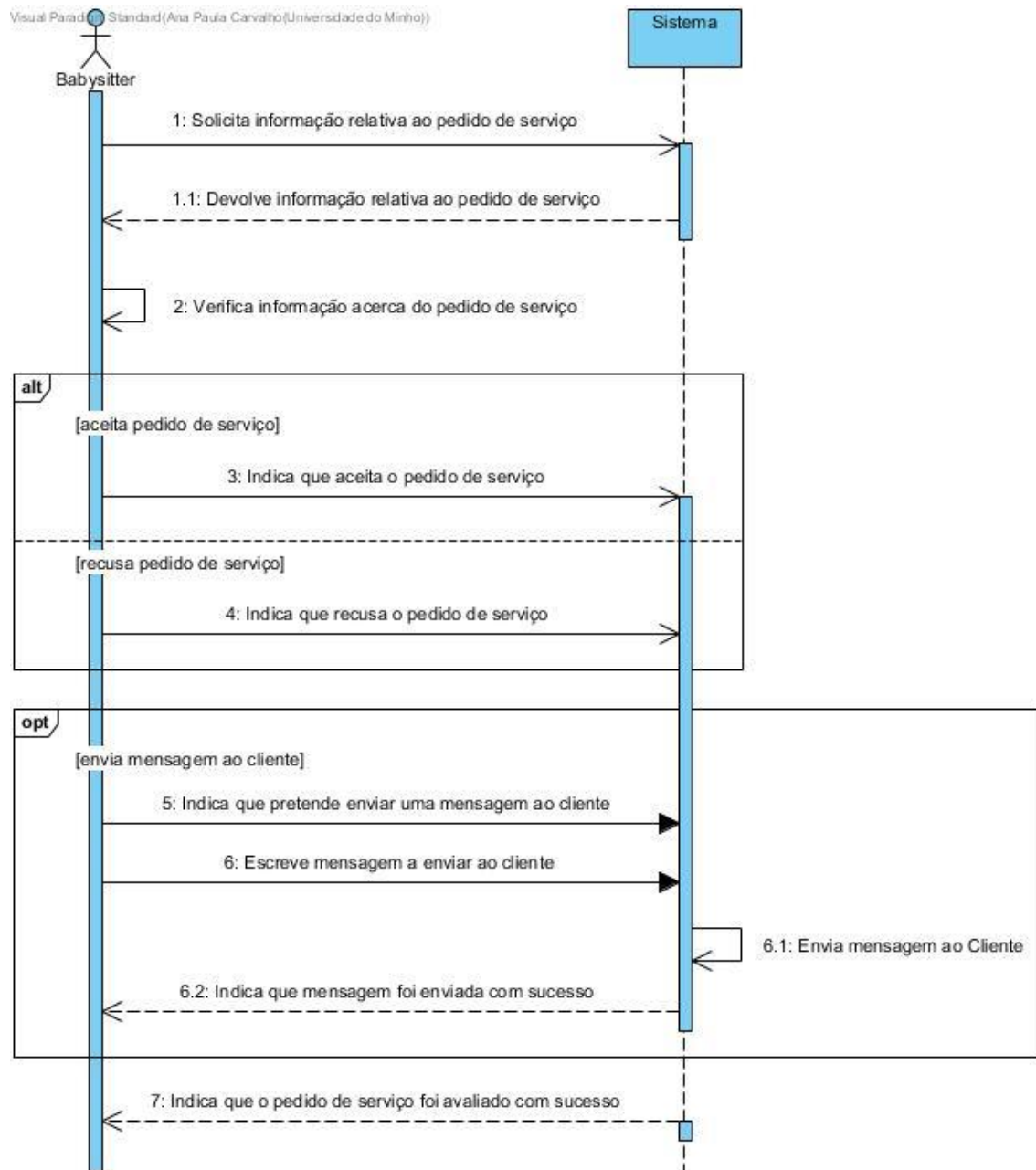


Figura 14 - DS de "Aceitar/recusar pedido de serviço"

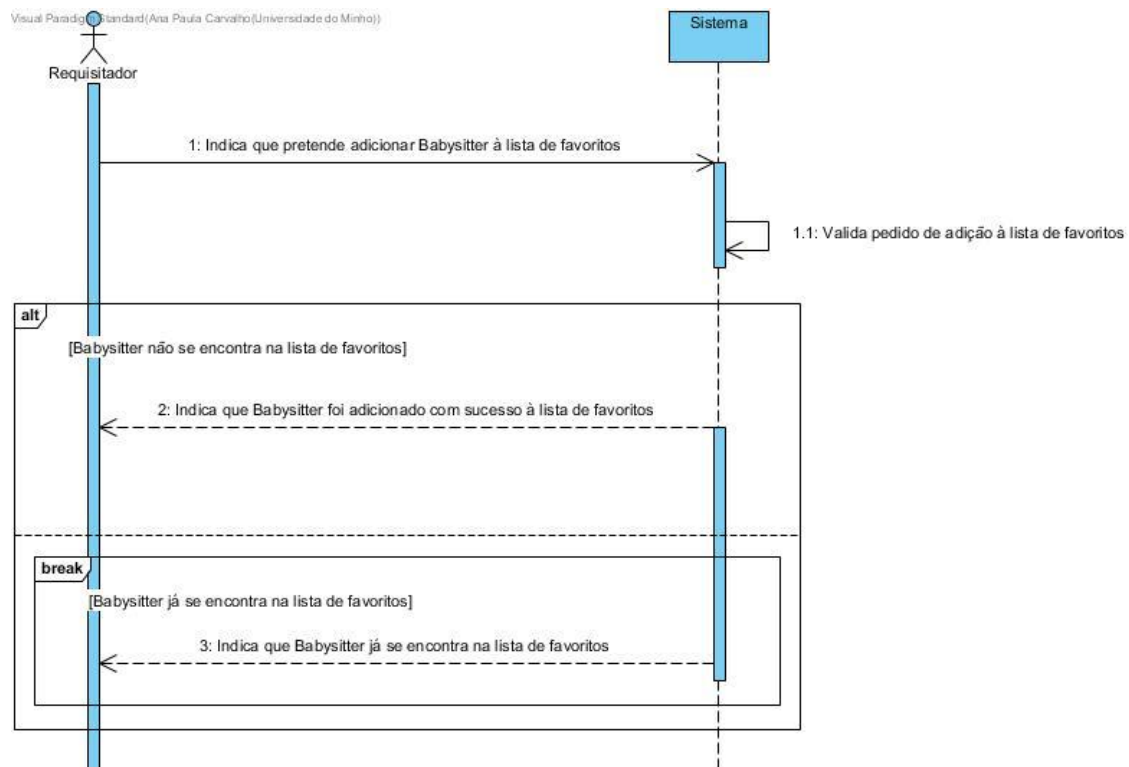


Figura 15 - DS de "Adicionar Babysitter à lista de favoritos"

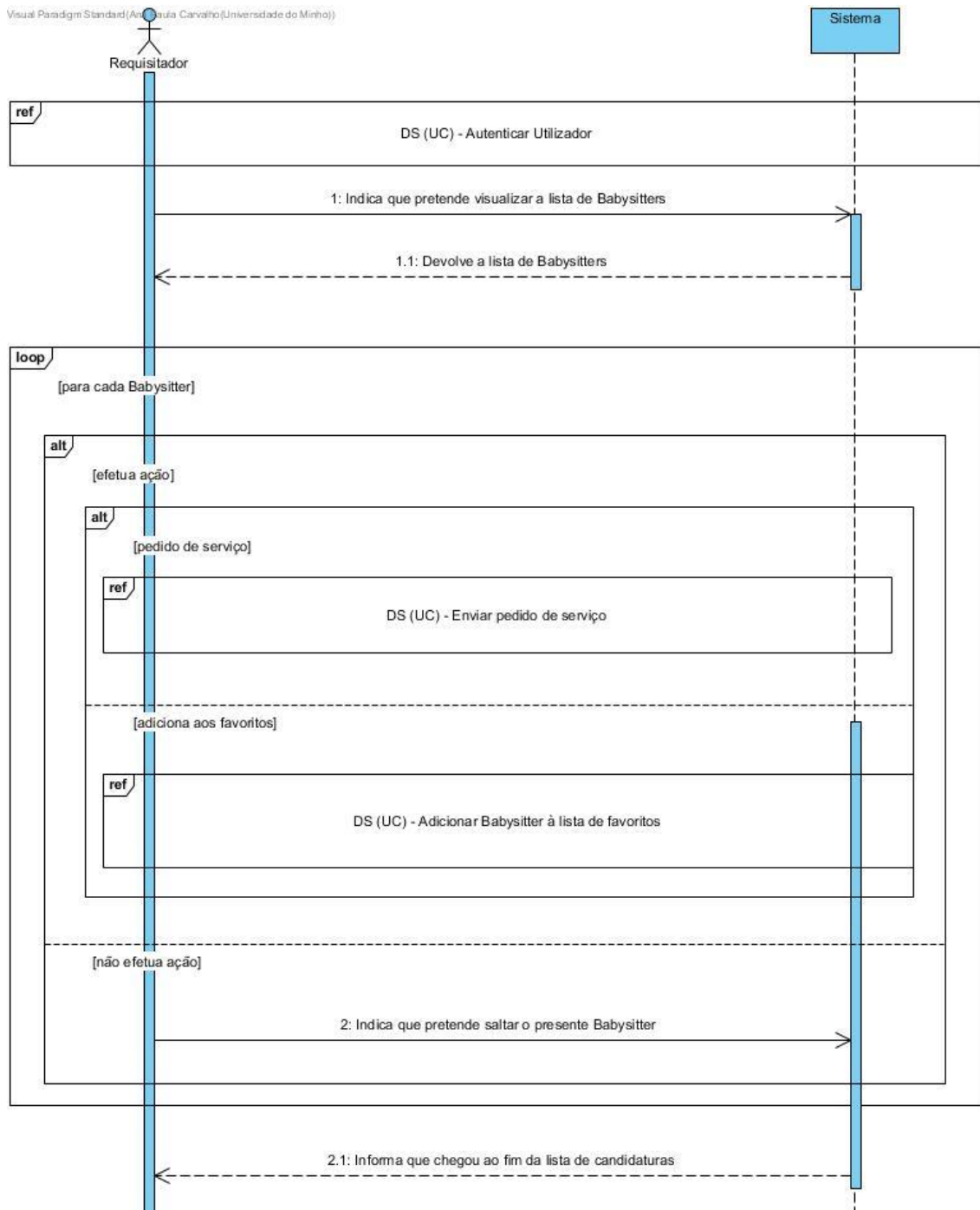


Figura 16 - DS de "Consultar lista de Babysitters"

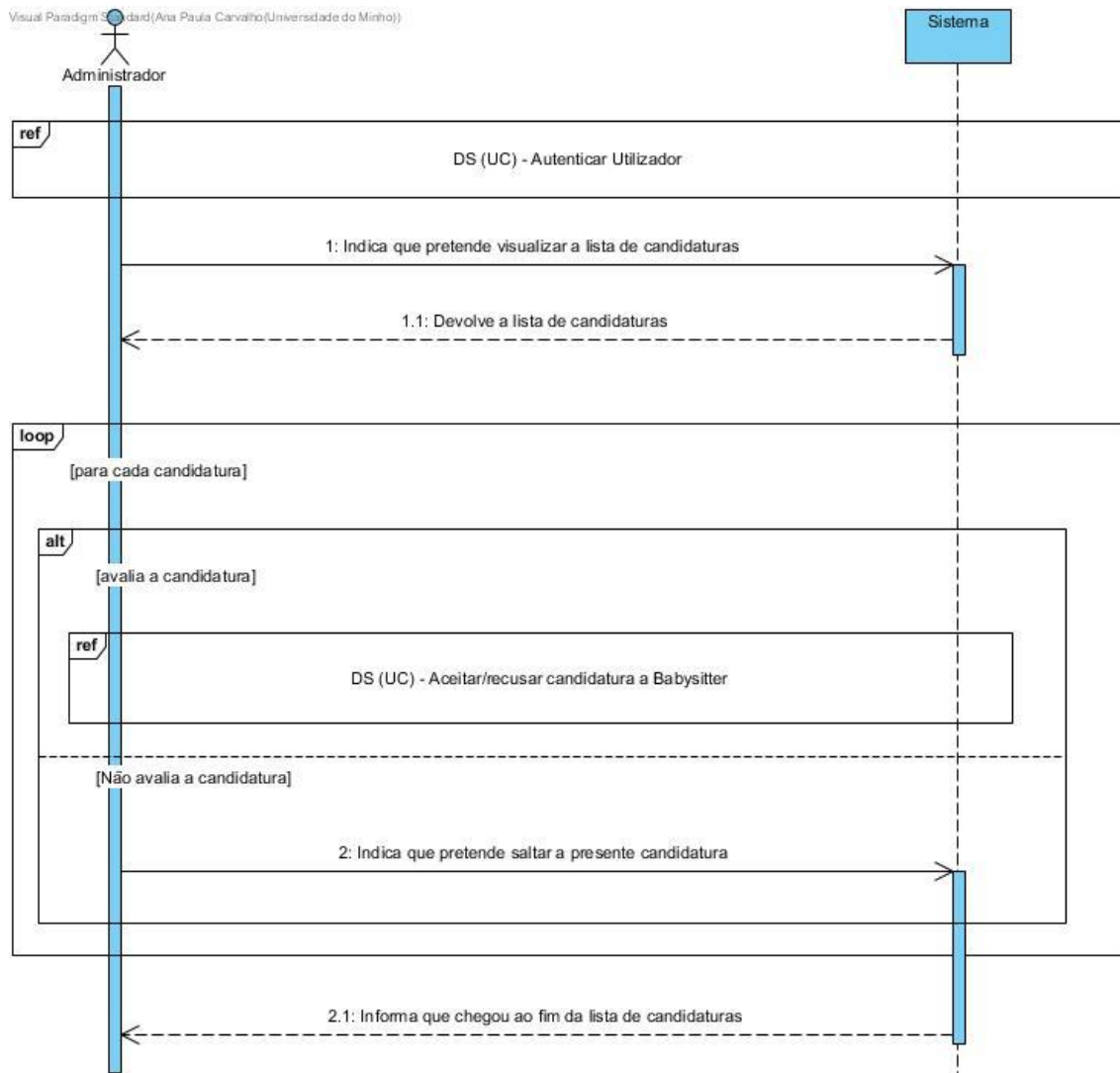


Figura 17 - DS de "Consultar lista de candidaturas a Babysitter"

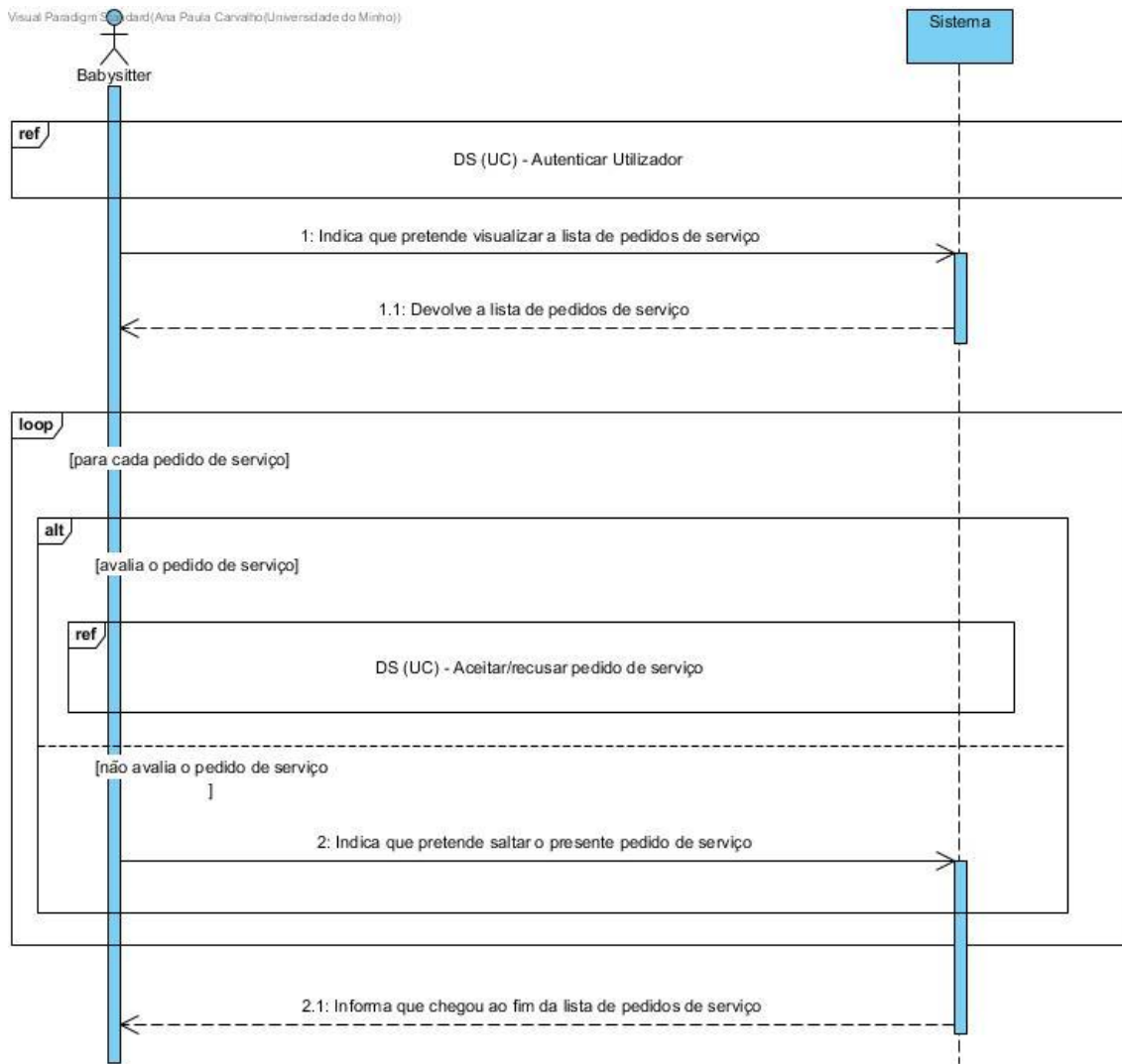


Figura 18 - DS de "Consultar lista de pedidos de serviço"

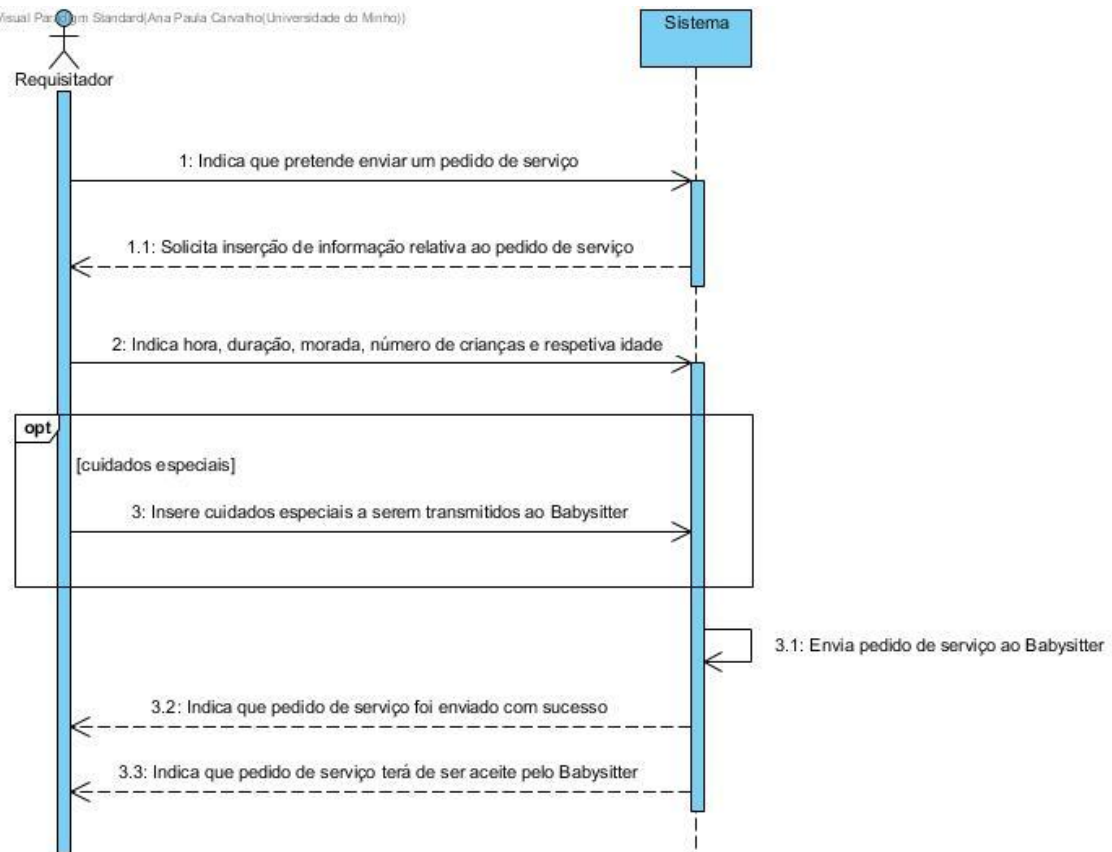


Figura 19 - DS de "Enviar pedido de serviço"

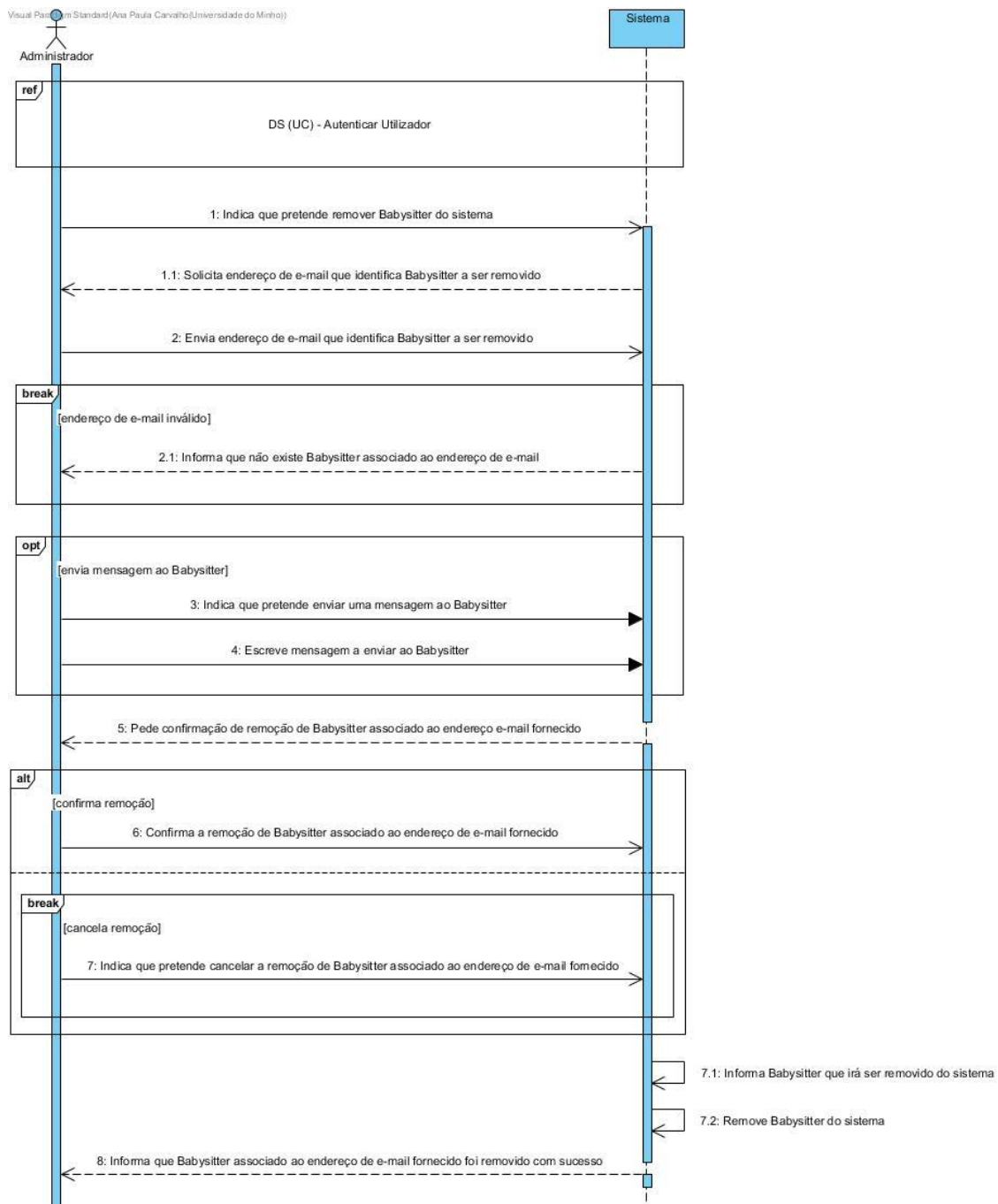


Figura 20 - DS de "Remover Babysitter do sistema"

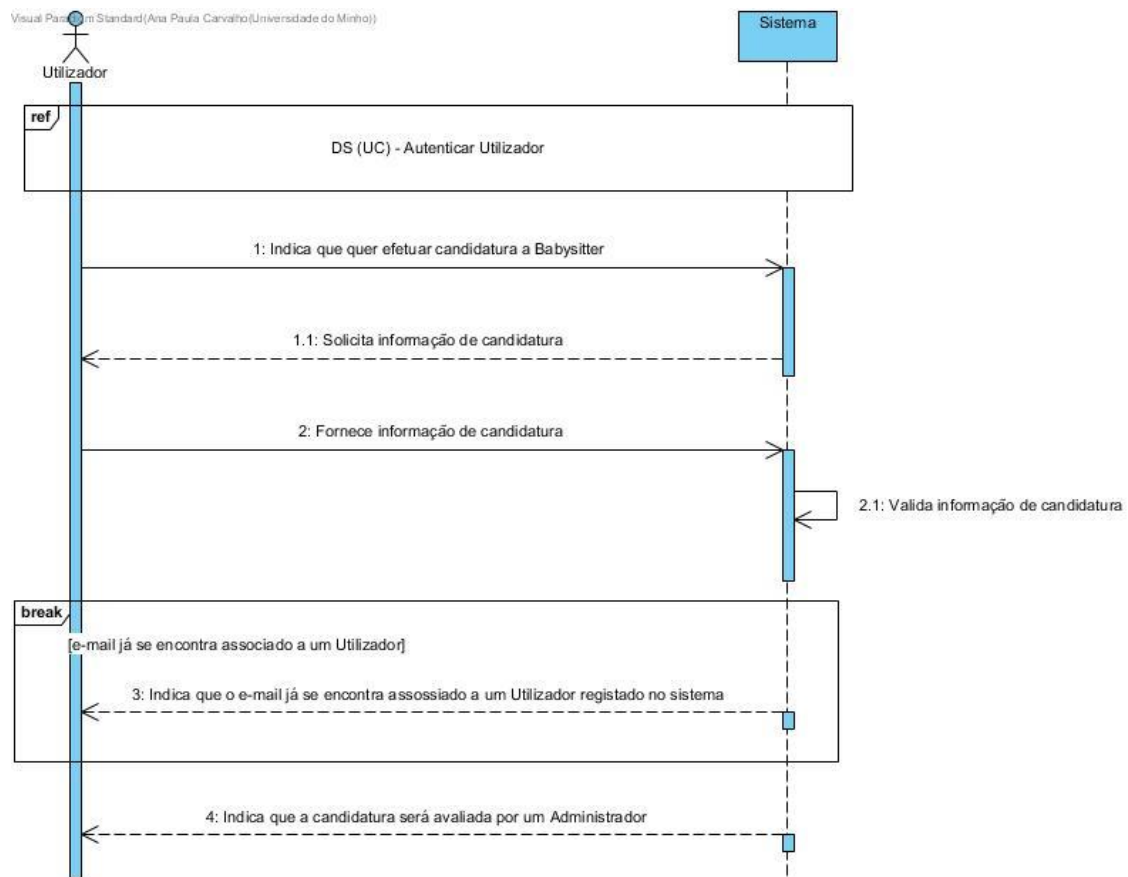


Figura 21 - DS de "Submeter candidatura"

3.1.4 Diagramas de Sequência de Subsistemas

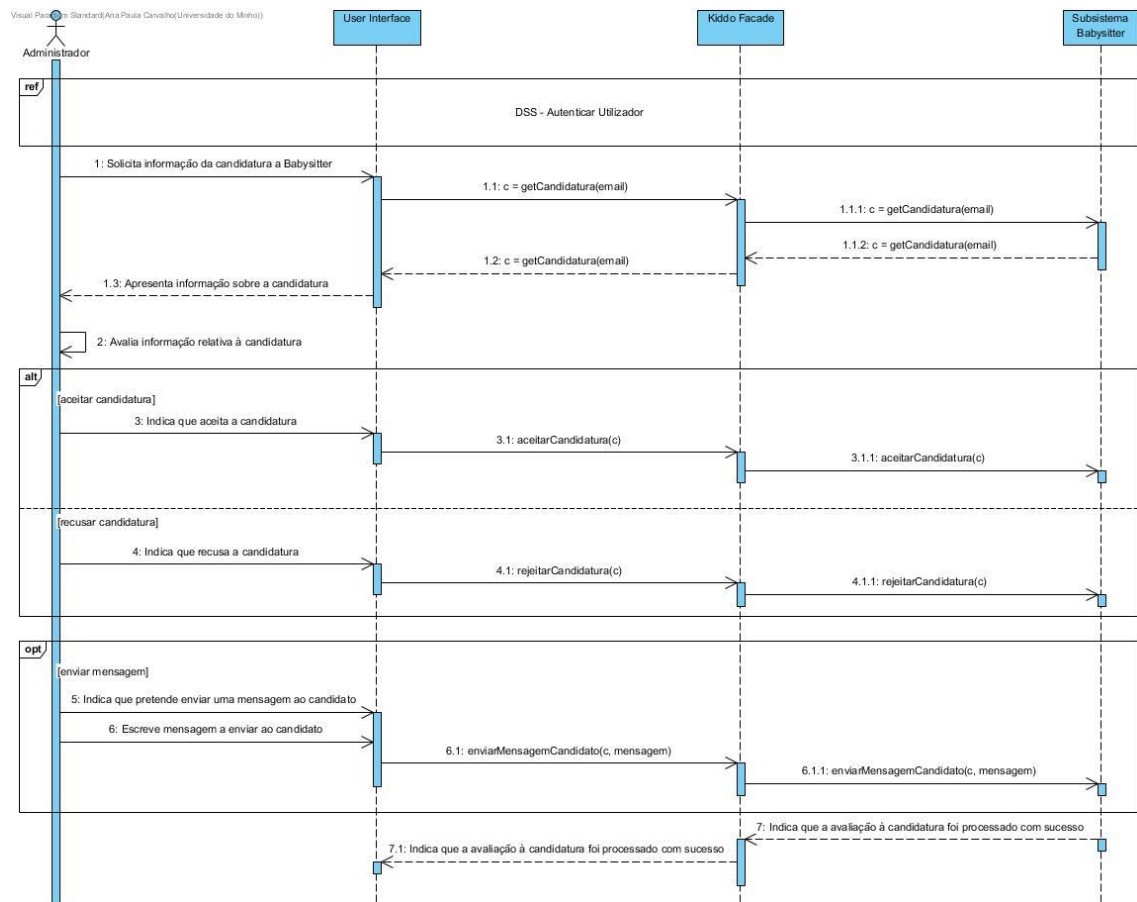


Figura 22 – DSS de “Aceitar/recusar candidatura a Babysitter”

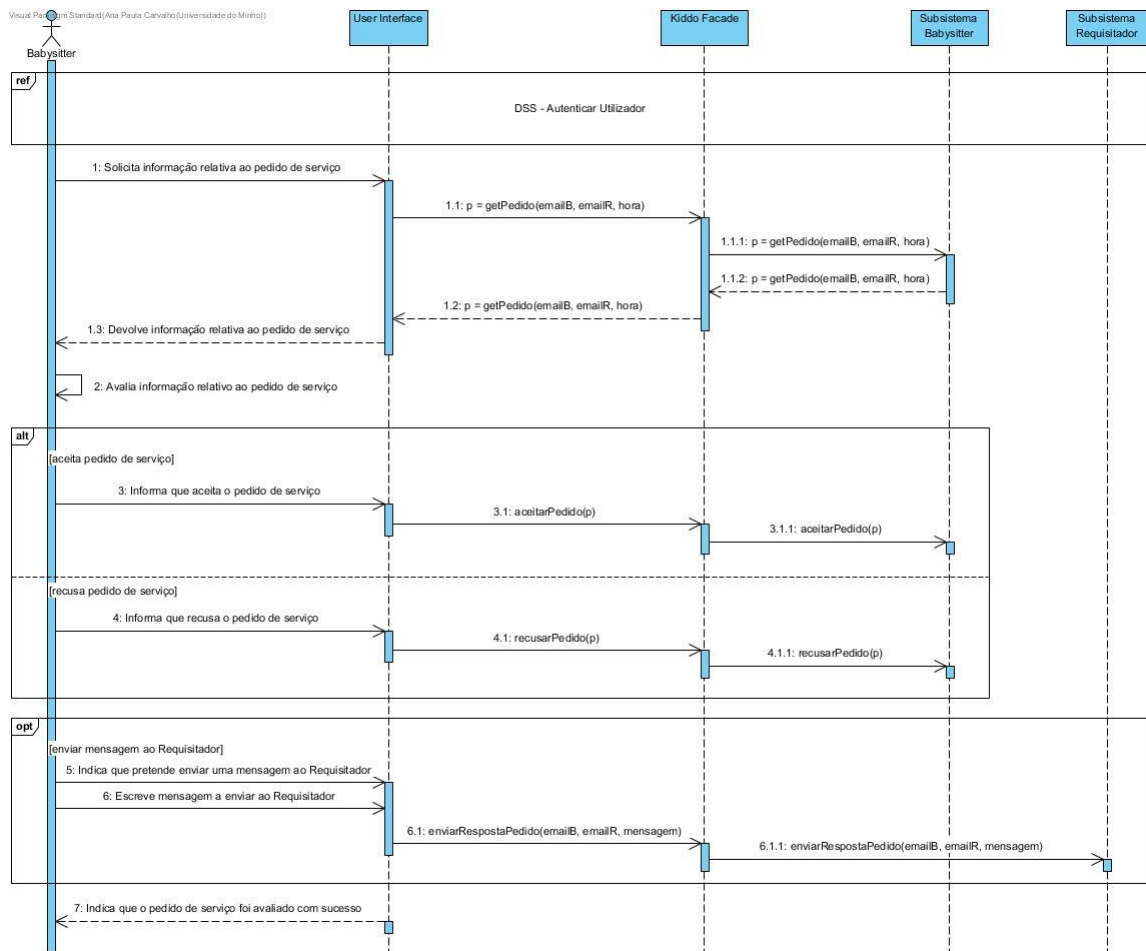


Figura 23 - DSS de "Aceitar/recusar pedido de serviço"

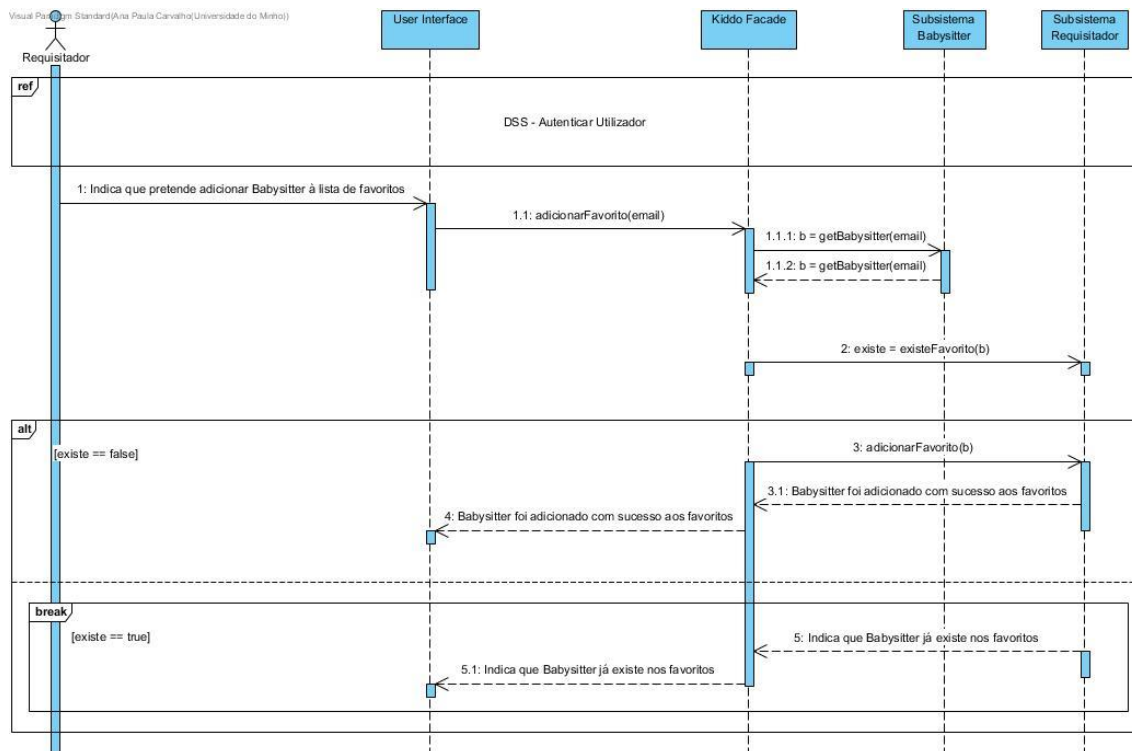


Figura 24 - DSS de "Adicionar Babysitter à lista de favoritos"

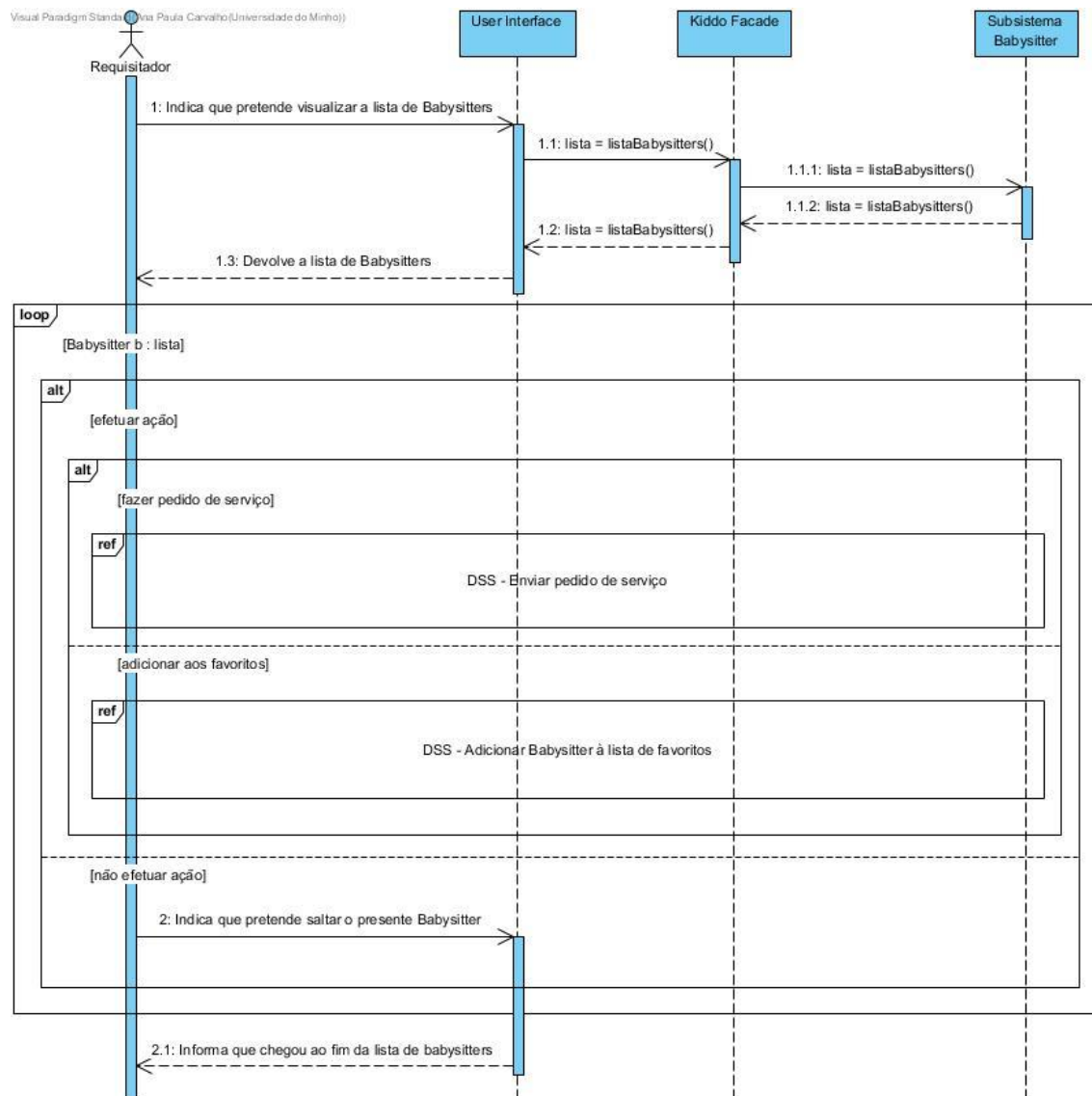


Figura 25 - DSS de "Consultar lista de Babysitters"

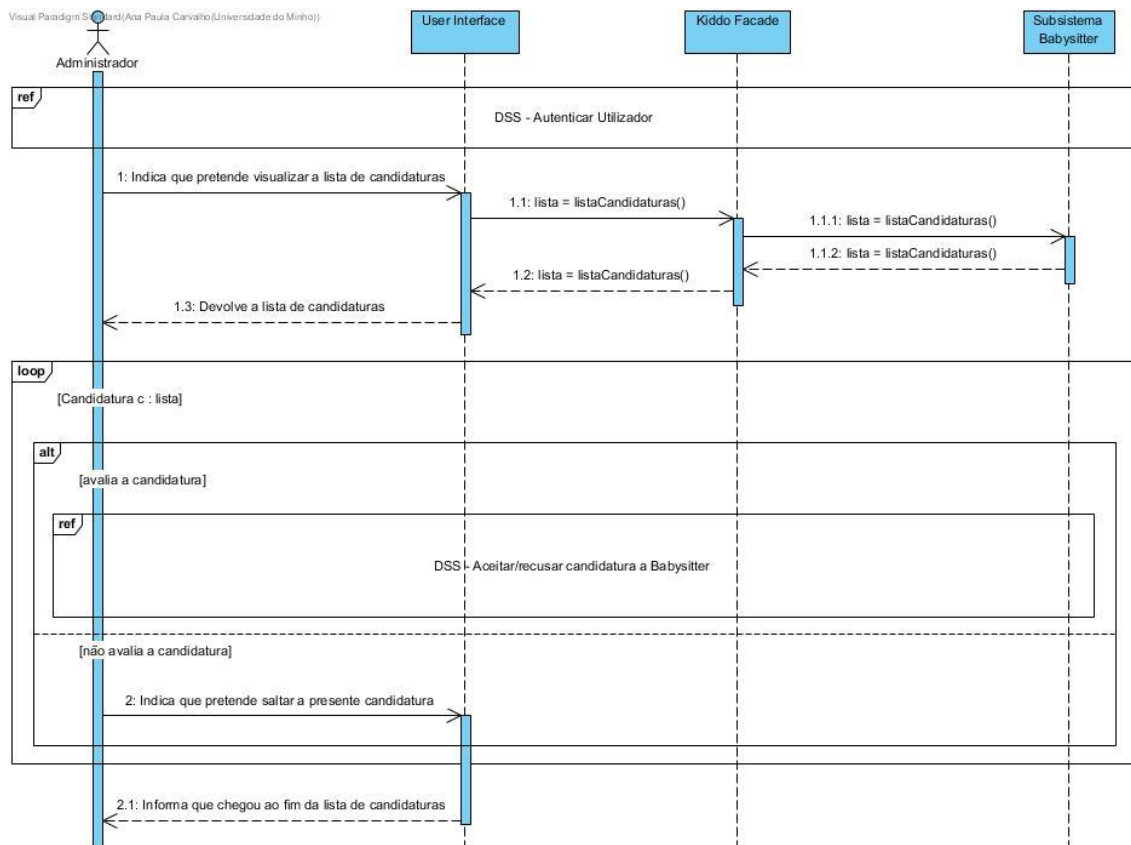


Figura 26 - DSS de "Consultar lista de candidaturas a Babysitter"

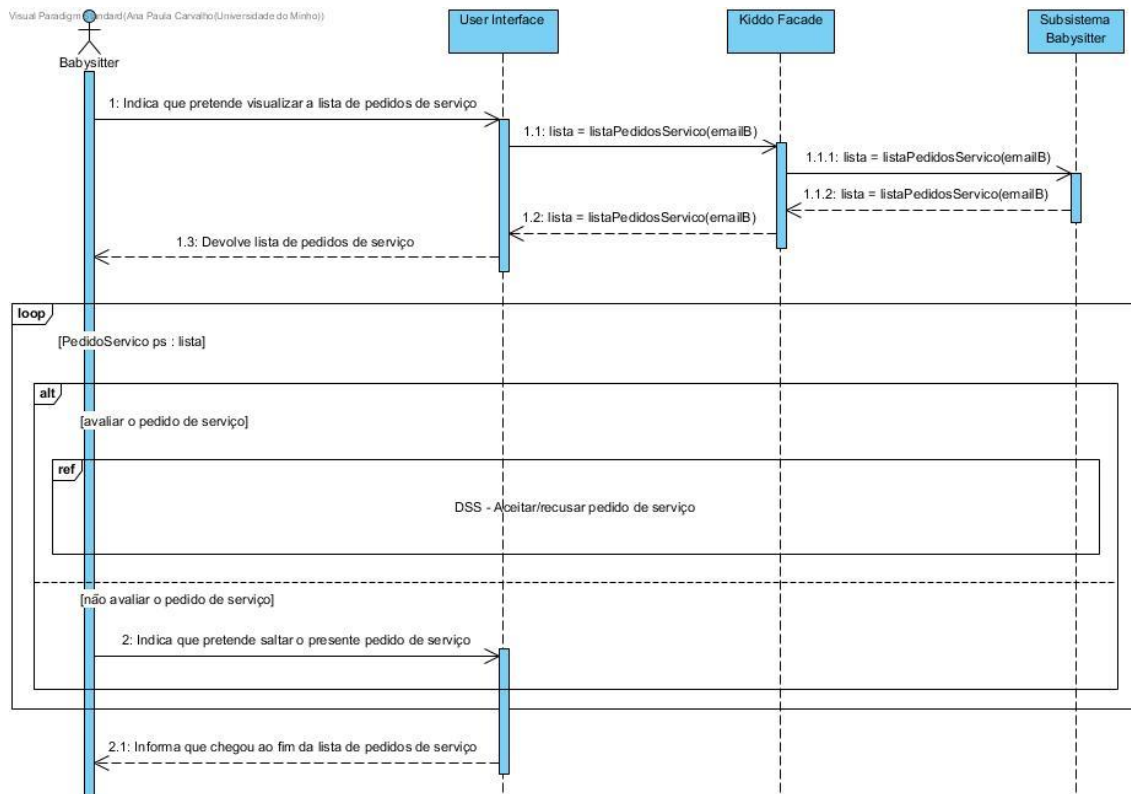


Figura 27 - DSS de "Consultar lista de pedidos de serviço"

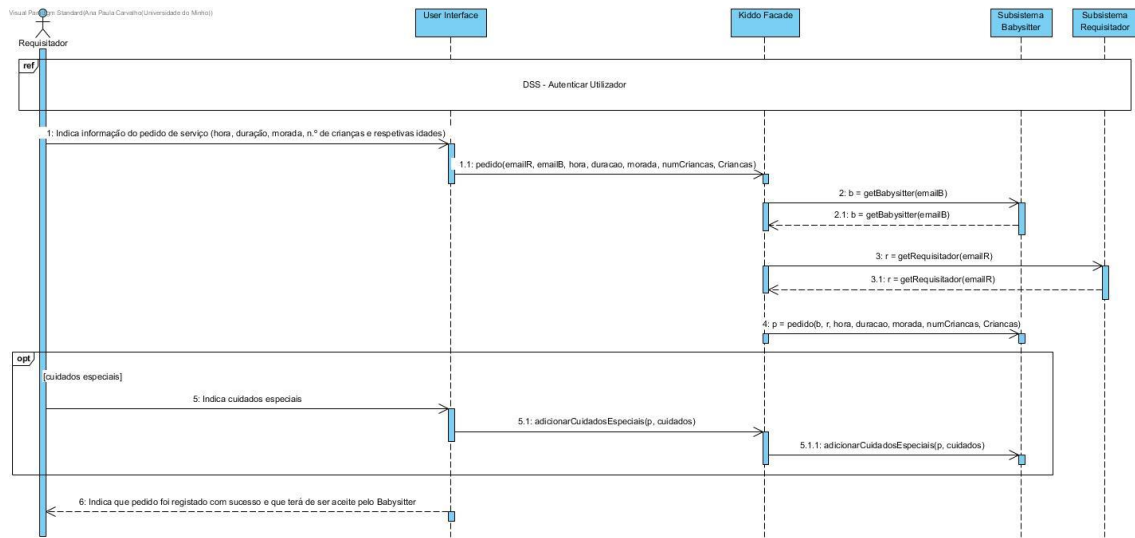


Figura 28 - DSS de "Enviar pedido de serviço"

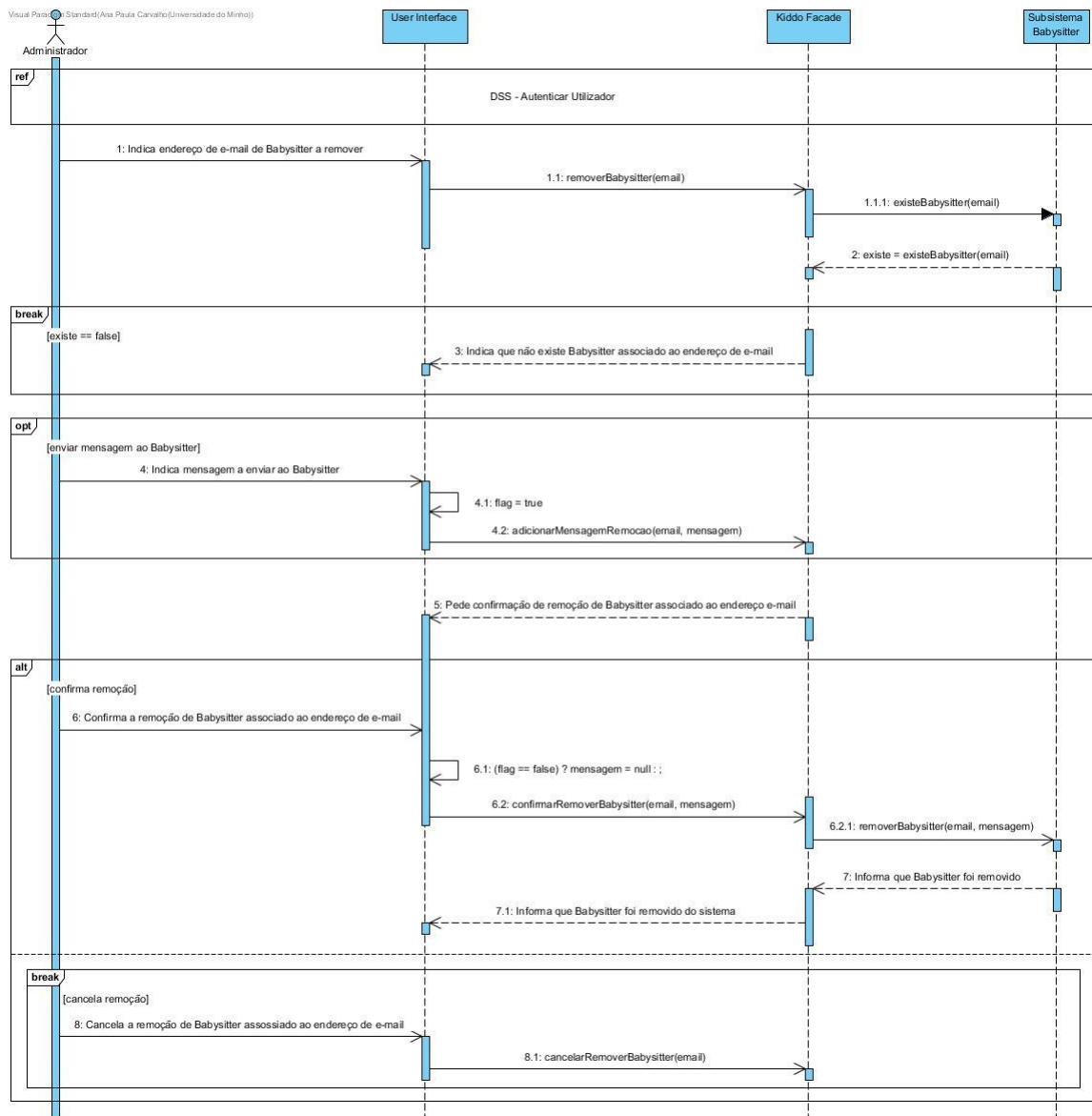


Figura 29 - DSS de "Remover Babysitter do sistema"

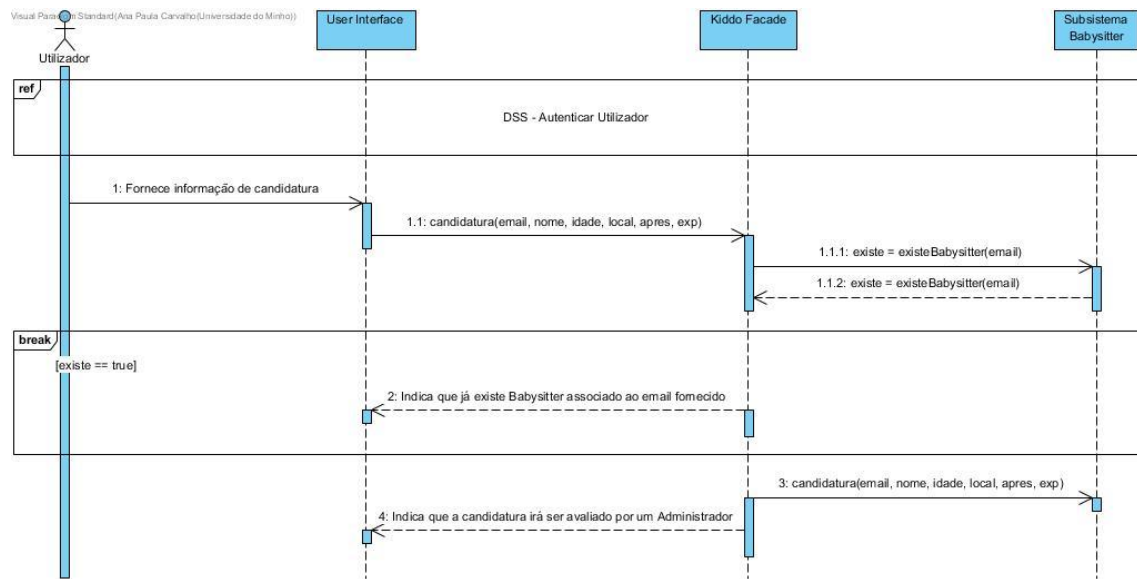


Figura 30 - DSS de "Submeter candidatura"

3.1.5 Diagramas de Atividade

Os diagramas de atividade são essencialmente gráficos de controlo de fluxo de uma atividade, significativos para a modelagem de aspetos dinâmicos do sistema. Revelam a sua utilidade demonstrando claramente o encadeamento de processos e, por conseguinte, facilitando a criação de *software* ao providenciar uma espécie de guião. Tudo isto contribui para que o programa corresponda ao exigido pelo cliente.

Devido à sua importância na produção de aplicações de qualidade, foram realizados alguns diagramas de atividades.

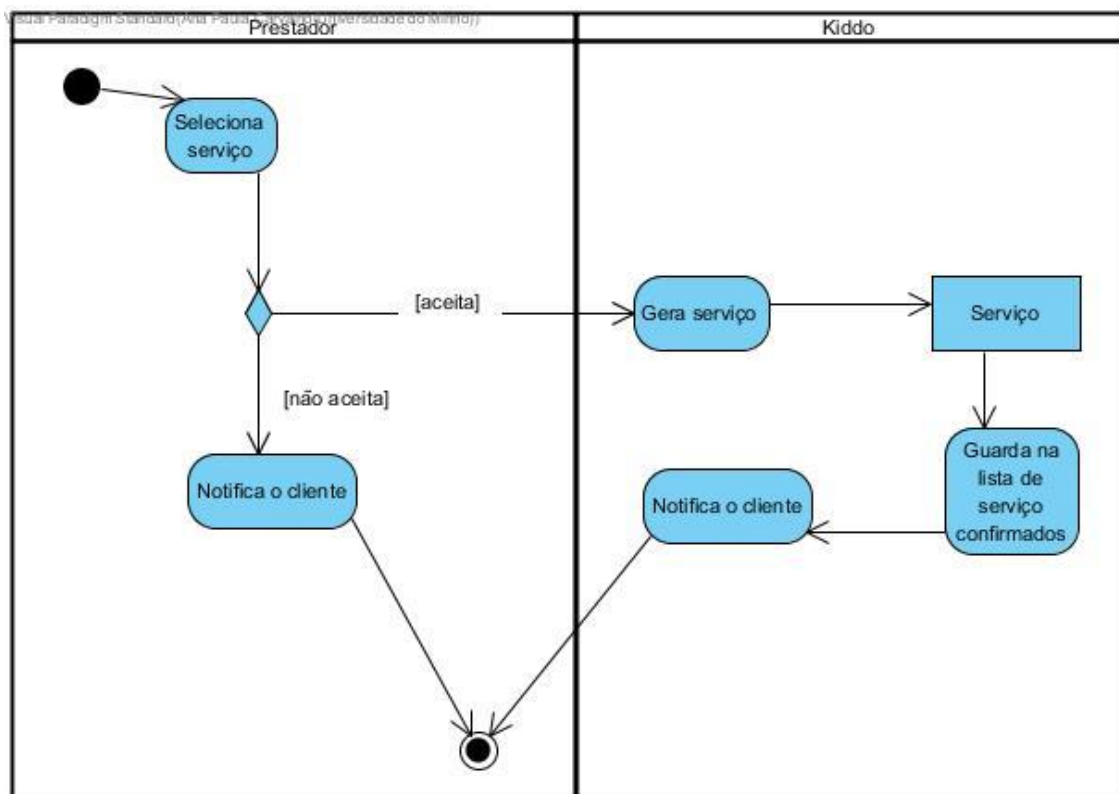


Figura 31 - DA de "Aceita pedido de serviço"

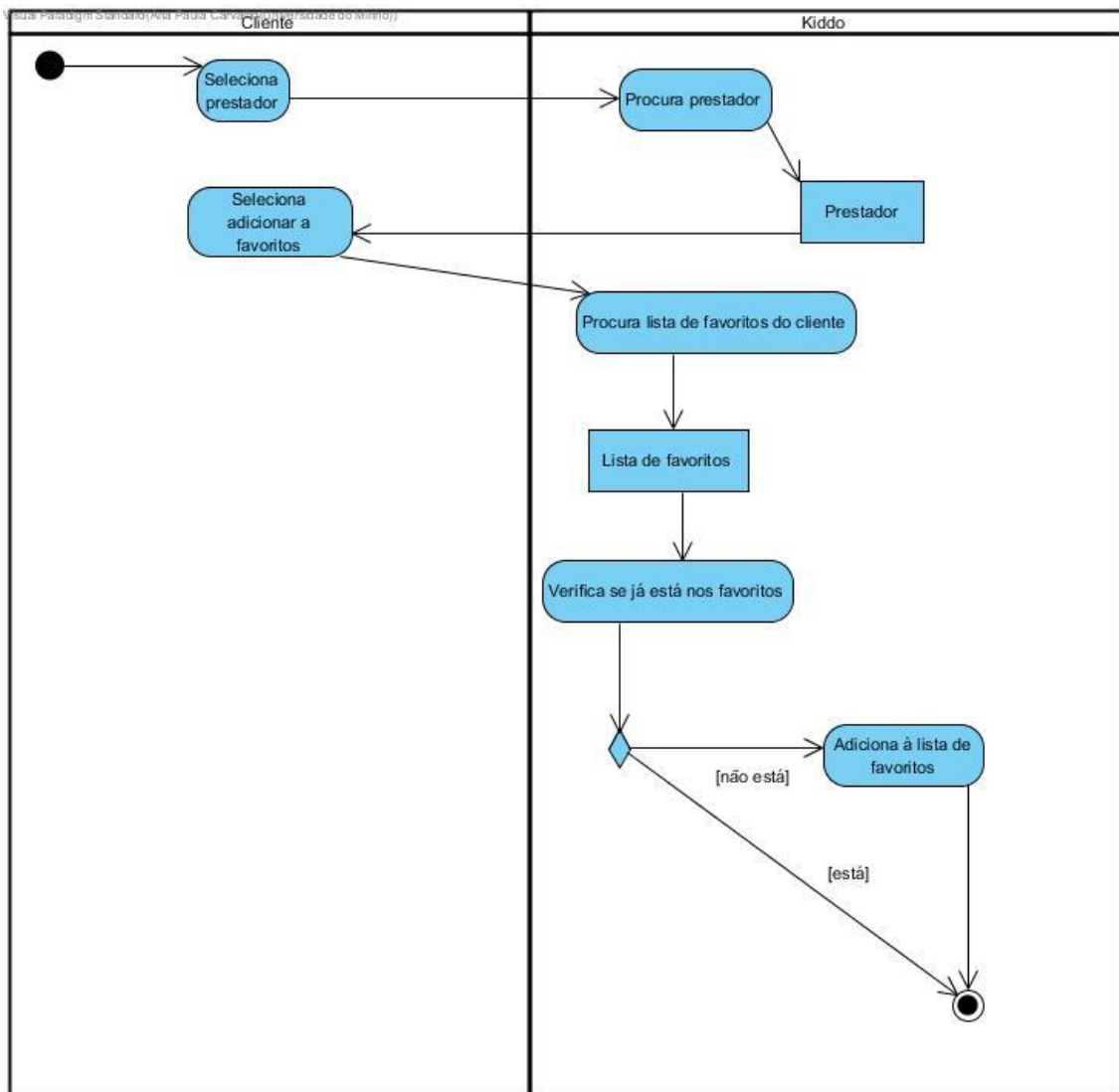


Figura 32 - DA de "Adiciona a lista de favoritos"

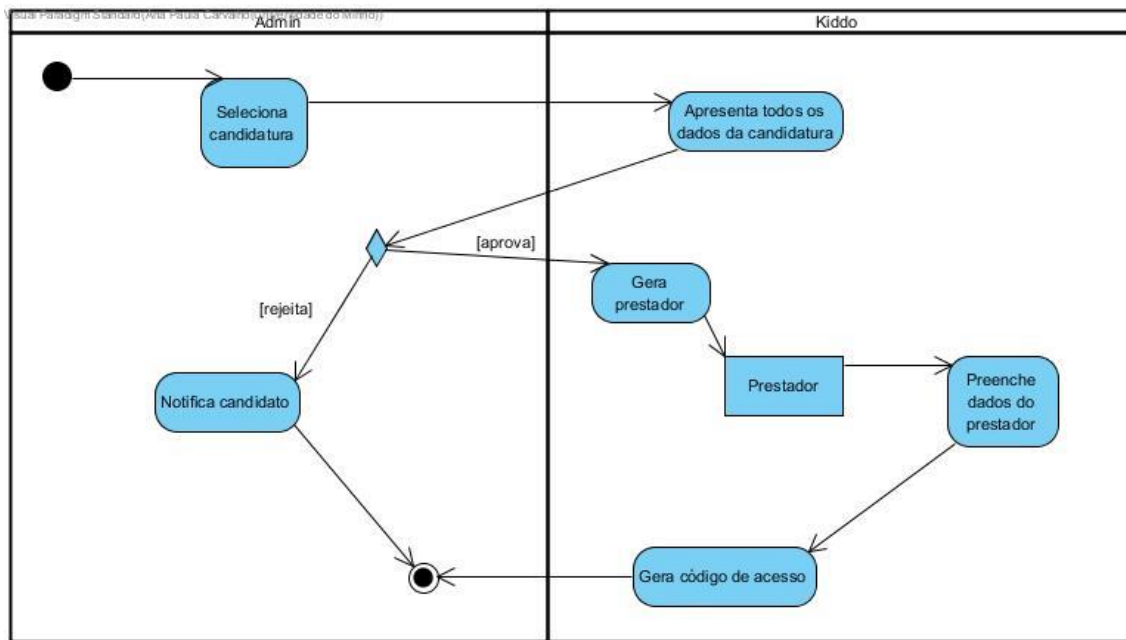


Figura 33 - DA de "Aprova babysitter"

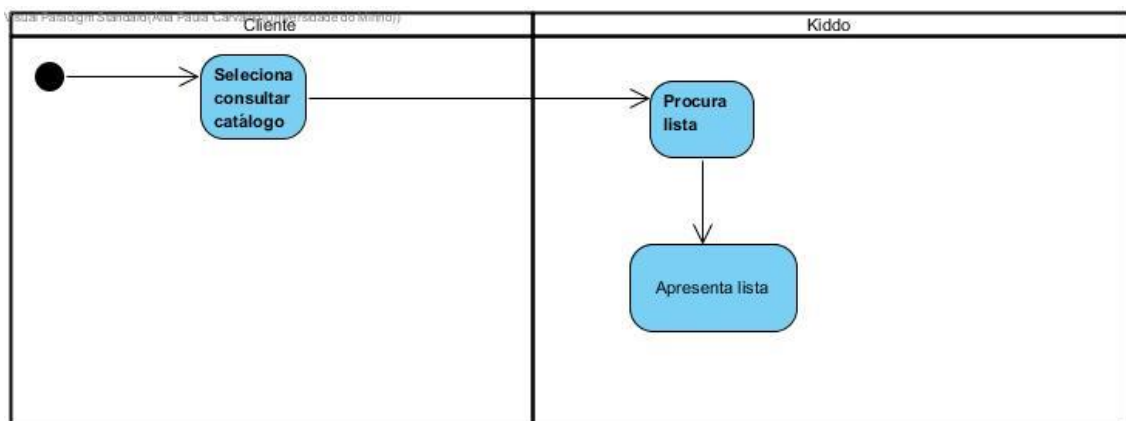


Figura 34 - DA de "Consulta catálogo de prestadores"

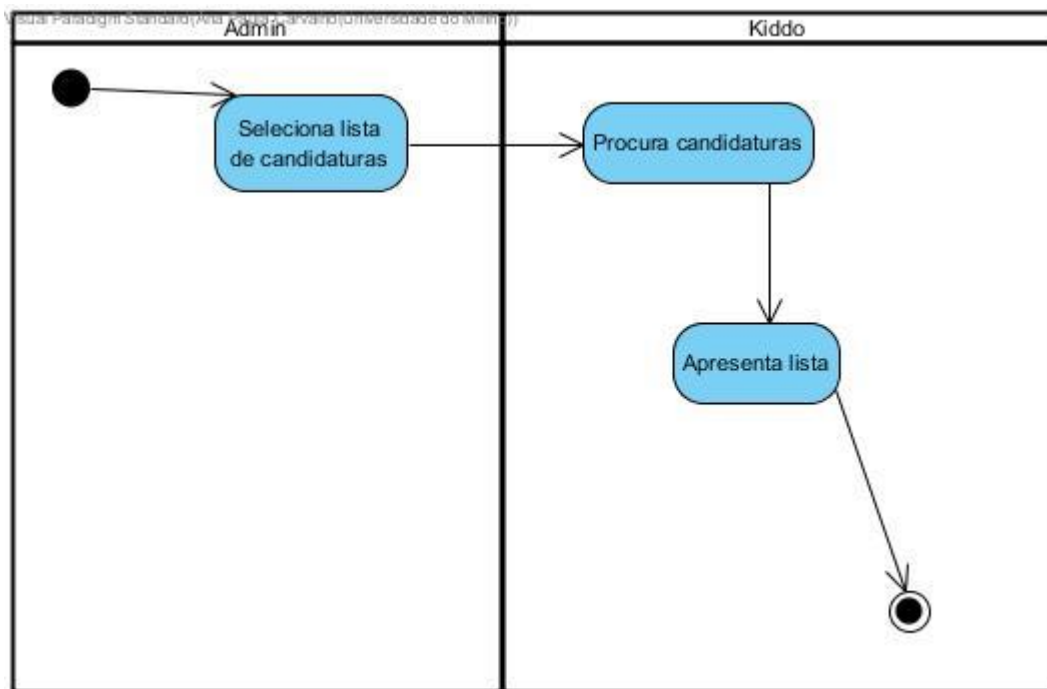


Figura 35 - DA de "Consulta lista de candidaturas"

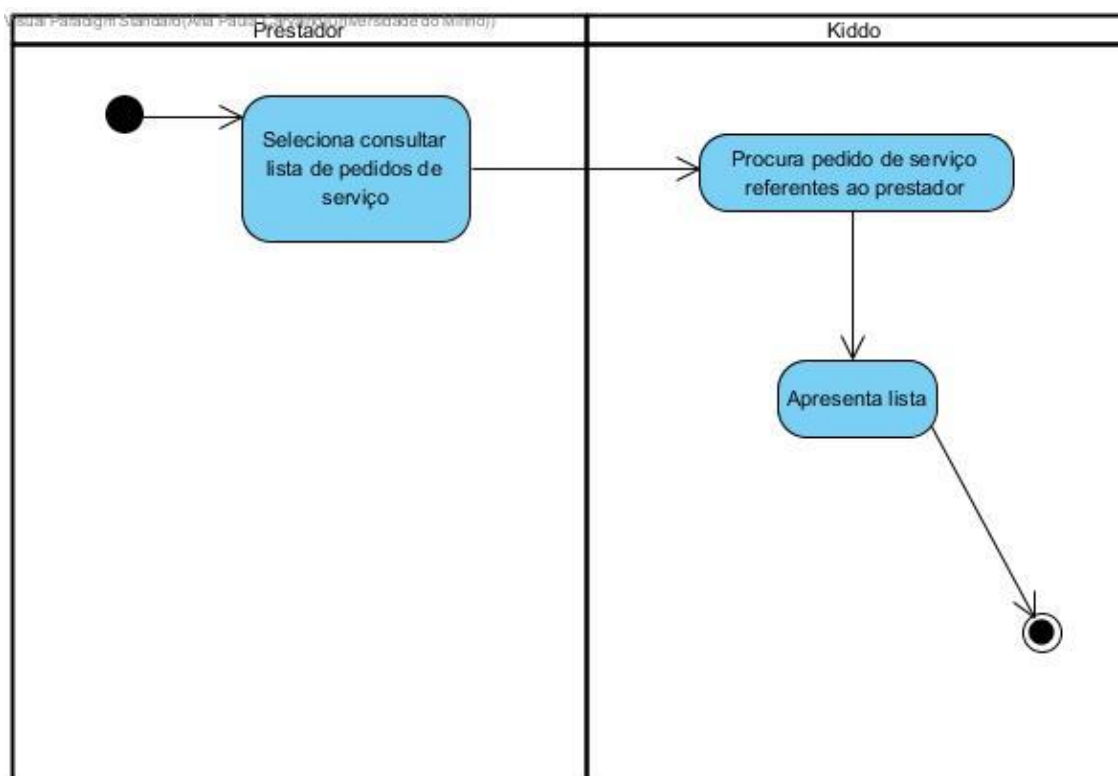


Figura 36 - DA de "Consulta lista de pedidos de serviço"

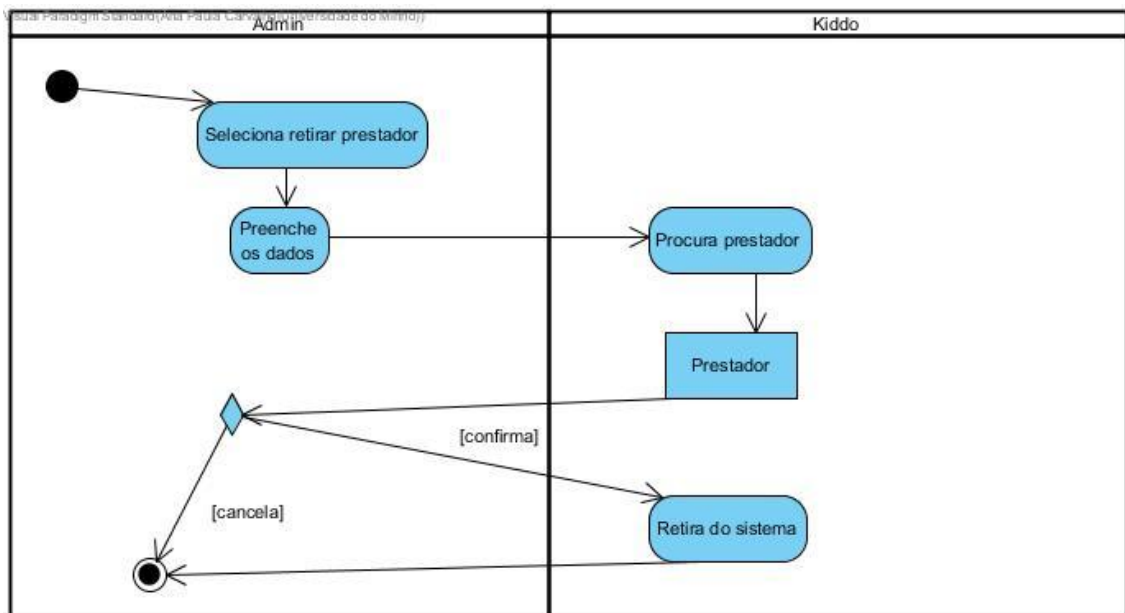


Figura 37 - DA de "Retira babysitter do sistema"

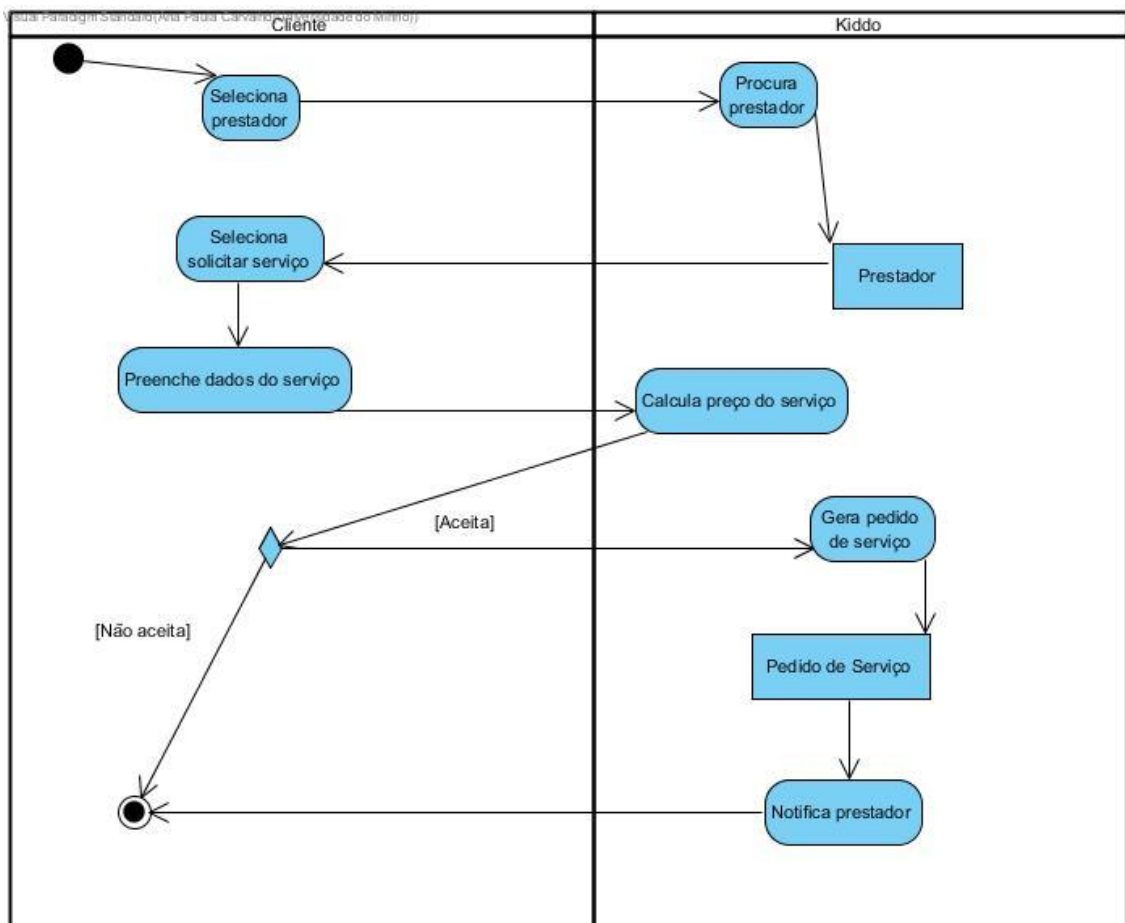


Figura 38 - DA de "Solicita serviço"

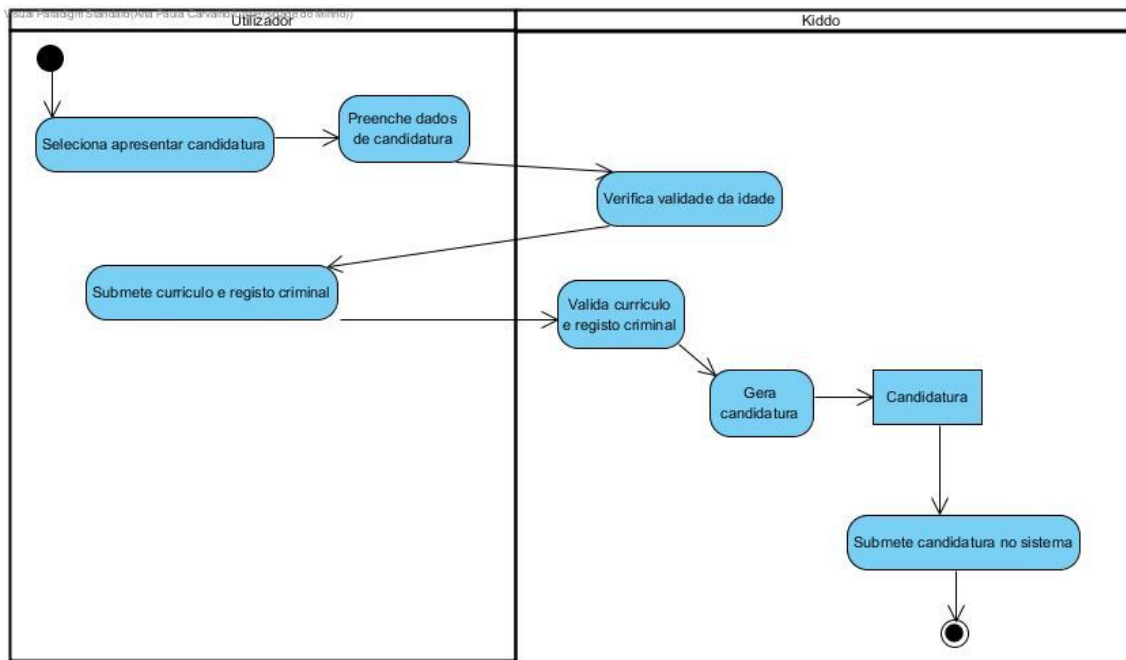


Figura 39 - DA de "Submete candidatura a babysitter"

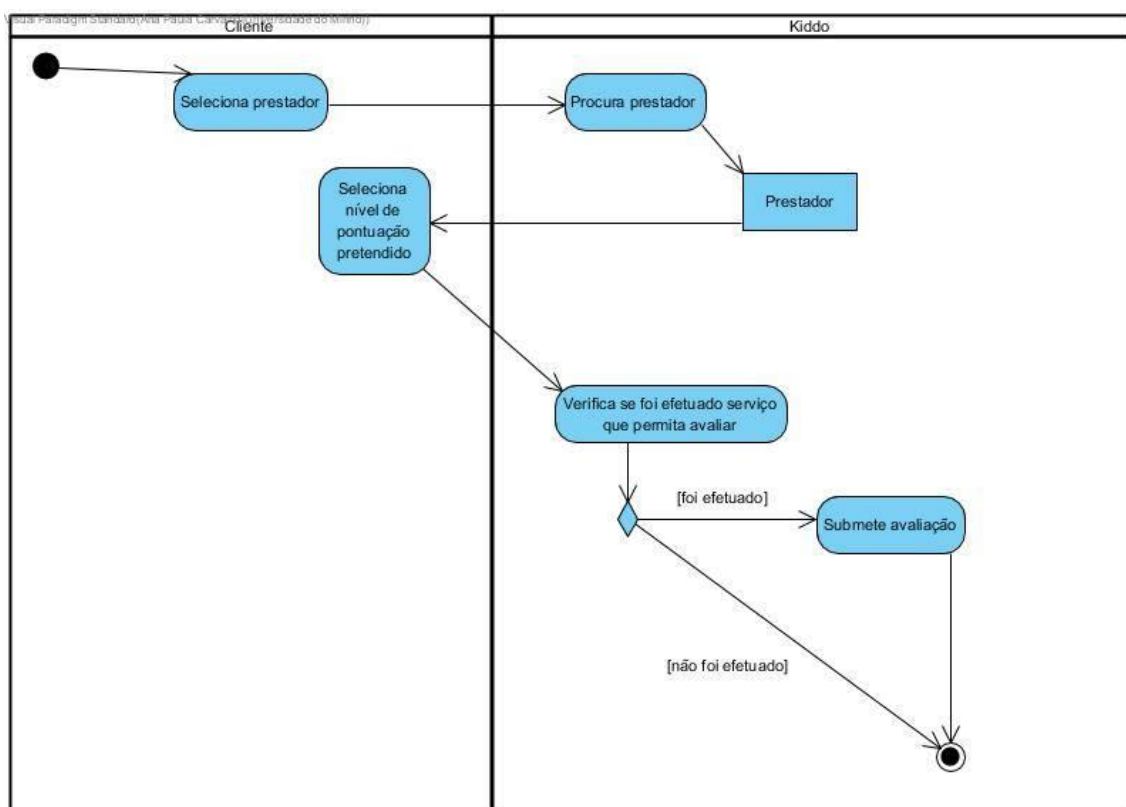


Figura 40 - DA de "Submete pontuação"

3.1.6 Modelo de Domínio

O Modelo de Domínio é construído para ilustrar os conceitos e relações entre estes, apresentando assim um modelo conceptual do problema em questão e nele podem ser visualizadas as entidades que participam no negócio da empresa, assim como os papéis e atributos que desempenham.

Os utilizadores podem ser *babysitters* ou requisitadores. *Babysitters* possuem um perfil onde são encontrados os seus vários atributos, assim como o requisitador que possui um perfil próprio. O requisitador pode consultar um catálogo de perfis de *babysitters*. O administrador aprova ou não a candidatura a *babysitter*. Existe também o pedido de serviço que contém vários atributos, serviço que contém uma fatura que está associado um valor.

Na imagem abaixo pode ser visto o Modelo de Domínio mencionado:

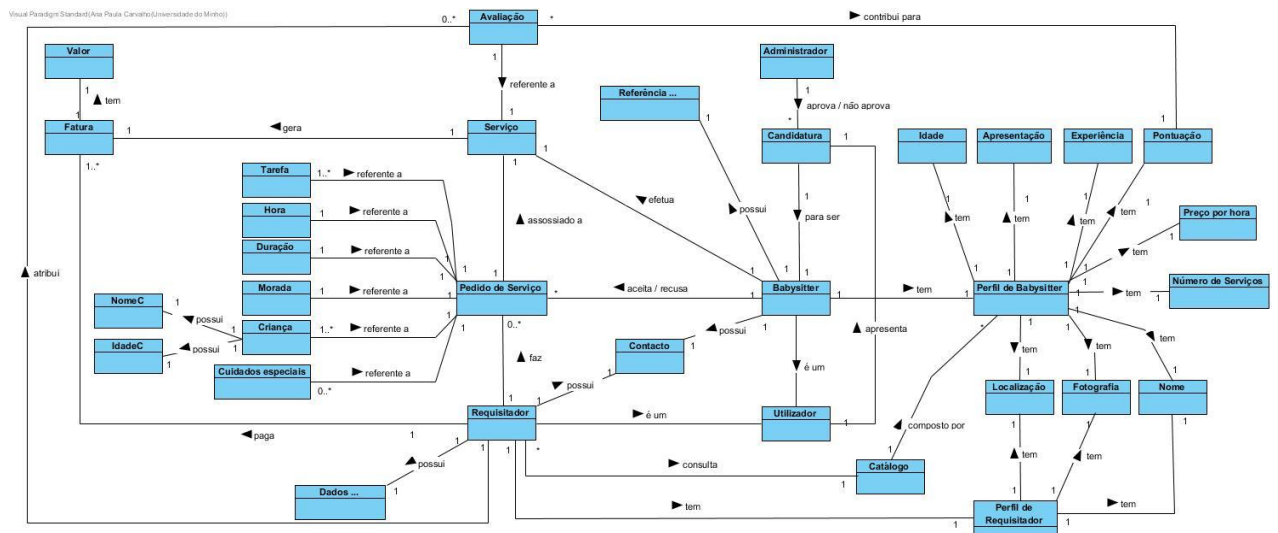


Figura 41 - Modelo de Domínio

3.1.7 Diagrama de Classes

De seguida, é apresentado o diagrama de classes. Este diagrama apresenta todas as classes, os seus respetivos atributos e métodos. Além disto, também são especificadas as relações que as classes têm umas com as outras.

Para a realização deste diagrama foram analisados os requisitos e o modelo de domínio.

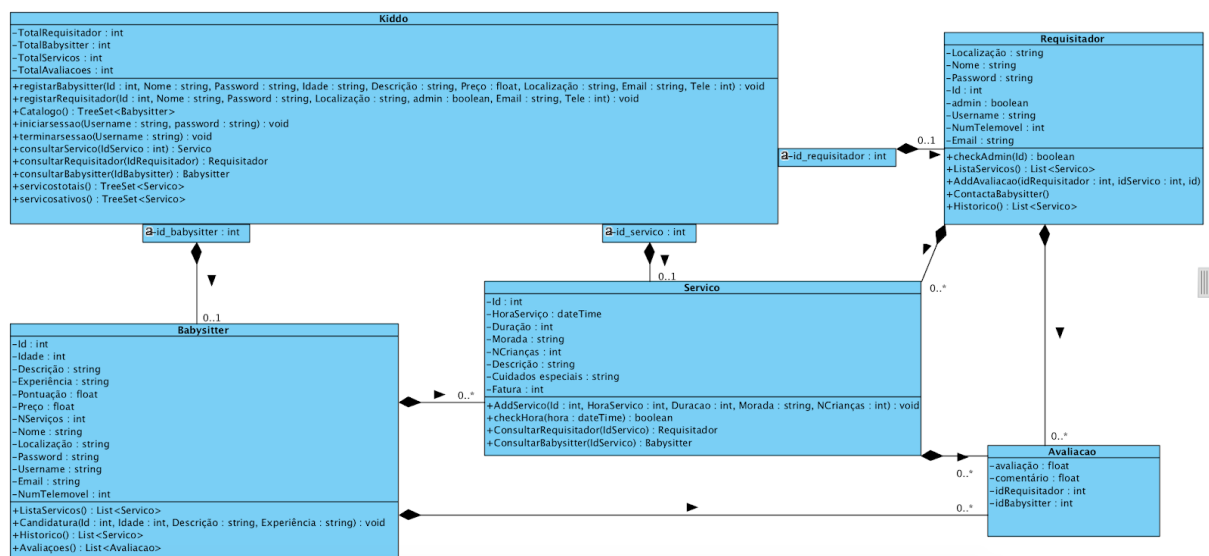


Figura 42 - Diagrama de Classes

3.1.8 Máquinas de Estado

Passando agora para a interface da nossa aplicação, será dividida em 2 partes como já explicado, ou seja, uma parte da nossa aplicação que só será para uso exclusivo às *babysitters*, para auxílio do serviço e a outra é o *website* onde se registam os utilizadores e onde se requisita o serviço. É assim importante esclarecermos detalhadamente como é que esta funciona e quais as funcionalidades disponíveis. Deste modo, elaboramos um conjunto de máquinas de estado que, de um certo modo, representam todas as formas de interação com a interface que o utilizador terá ao seu dispor no momento de utilização da aplicação.

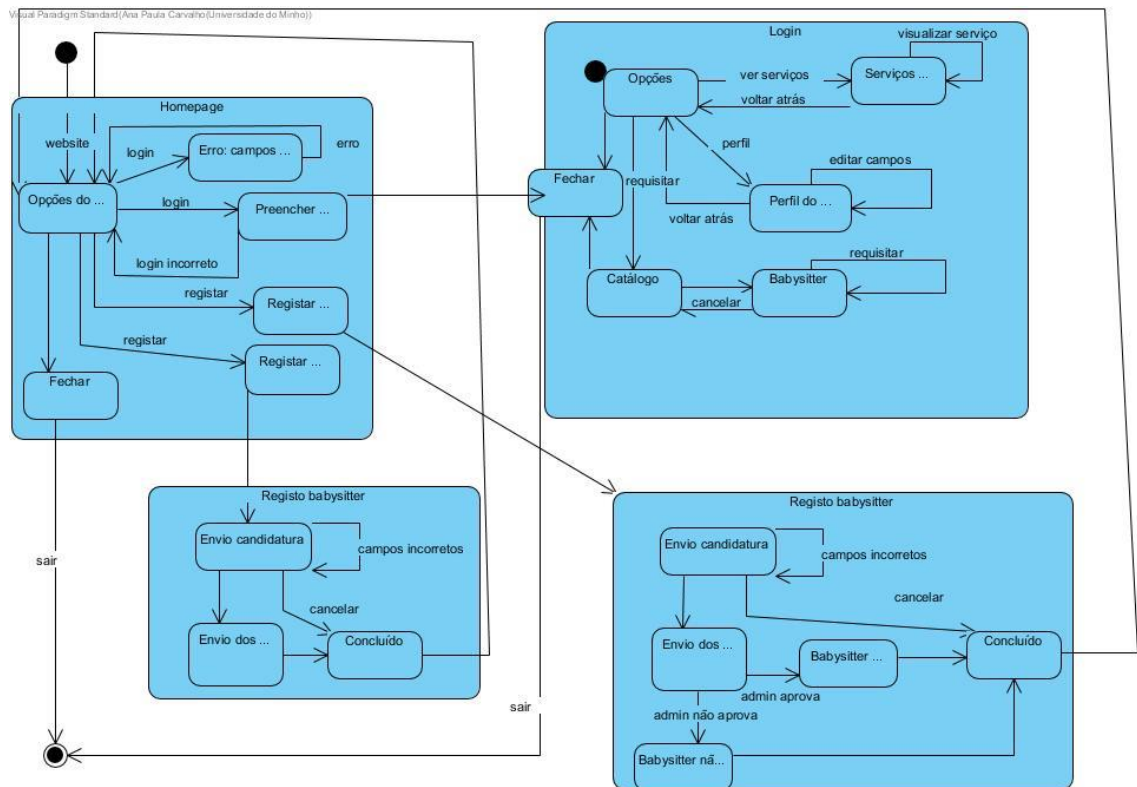


Figura 43 - Máquina de Estado (website)

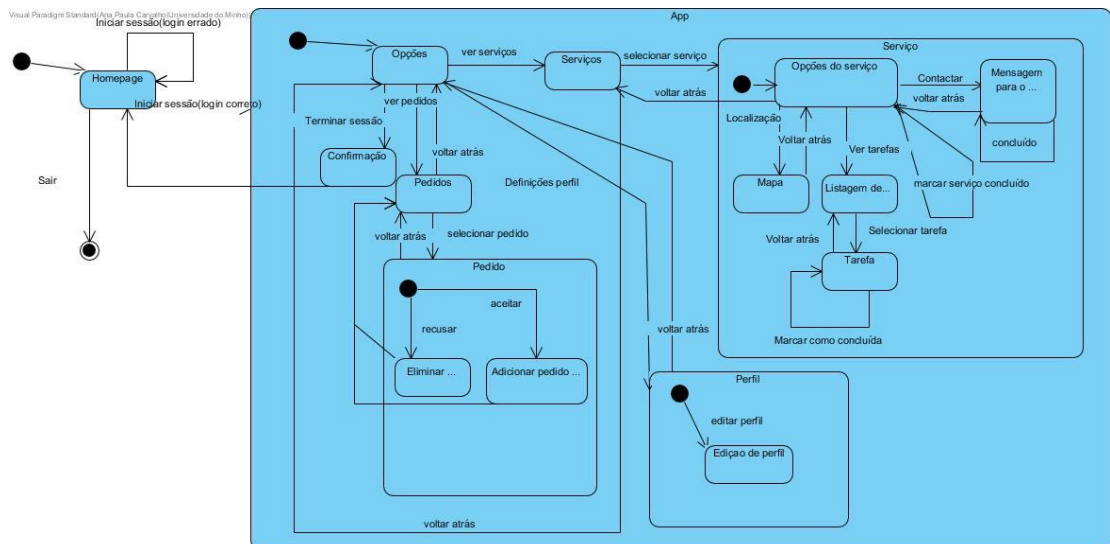


Figura 44 - Máquina de Estado (app)

4. Base de Dados

4.1. Modelo Conceptual

4.1.1 Diagrama do Modelo Conceptual

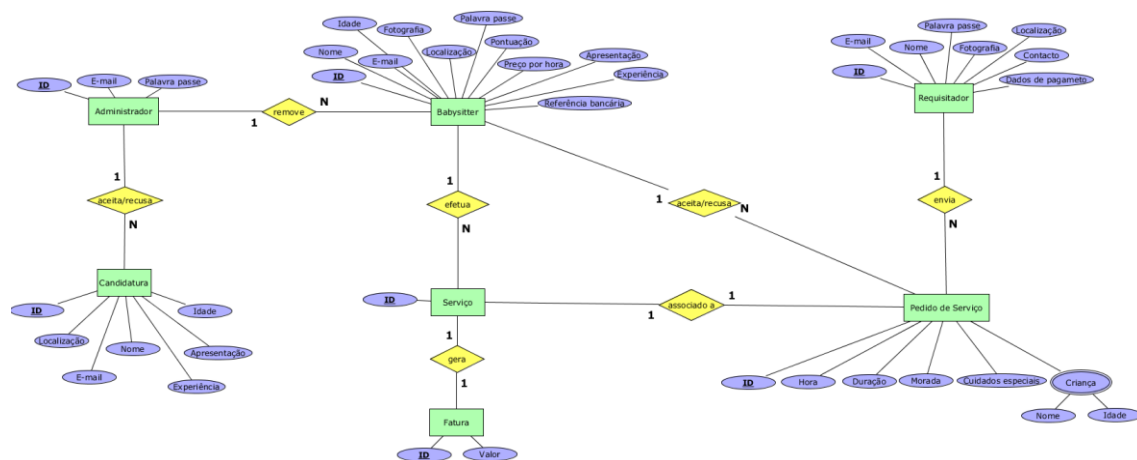


Figura 45 - Modelo Conceptual da Base de Dados

4.1.2 Identificação de Identities

Uma entidade é um objeto que tem uma existência própria e é distinguível de outros objetos pela sua relevância no sistema e sobre o qual é necessário armazenar informação. Assim sendo e analisando os requisitos enumerados anteriormente concluímos que a nossa base de dados teria sete entidades: Administrador, Candidatura, *Babysitter*, Serviço, Fatura, Pedido de Serviço e Requisitador.

Nome da Entidade	Descrição	Sinónimos
Administrador	Gestor do sistema que controla a inserção, remoção de babysitter e edição dos seus dados	-
Candidatura	Proposta enviada por um utilizador não registado para ser associado ao sistema como babysitter	Proposta
Babysitter	Utilizador que pretende usar a aplicação para receber pedidos de clientes	Prestador, ama
Serviço	Utilidade facultada por uma babysitter a um requisitador	-
Fatura	Documento gerado a partir de um serviço que foi efetuado	
Pedido de Serviço	Solicitação dos serviços uma babysitter por requisitador	Solicitação
Requisitador	Utilizador que pretende usar o site para solicitar serviços ou avaliar babysitters	Cliente

4.1.3 Identificação dos Relacionamentos

Após a identificação das entidades da nossa SBD, é necessário entender também como as entidades se relacionam entre si. Mais uma vez, verificando os requisitos impostos, torna-se possível não só compreender esses relacionamentos como também a suas cardinalidades.

4.1.4 Identificação e Associação de Atributos

De seguida, refletiu-se sobre a informação que seria necessário reter de cada entidade. Detetaram-se todos os atributos de cada entidade e identificaram-se os seus tipos.

4.1.5 Determinação do Domínio dos Atributos

Entidade: Administrador	
Atributo	Domínio
Id	int
Email	string
Password	string

Entidade: Candidatura	
Atributo	Domínio
Id	string
Email	string
Nome	string
Idade	int
Localização	string
Apresentação	string
Experiência	string
Aceite	boolean
BabysitterId	int
AdminId	int

Entidade: Babysitter	
Atributo	Domínio
Id	int
Email	string
Password	string
Nome	string
Idade	int
Foto	string
Localização	string
Pontuação	int
Preço/Hora	double
Apresentação	string
Referência Bancária	int
Contacto	int
Experiencia	string

Entidade: PedidoServico	
Id	int
RequisitadorId	int
BabysitterId	int
DataHora	data
Duracao	tempo
Morada	string
CuidadosEspeciais	string
Servicold	int

Entidade: Requisitador	
Id	Int
Email	string
Password	String
Nome	String
Foto	String
Localizacao	String
Contacto	Int
Pagamento	Int

Entidade: Fatura	
Id	Int
Valor	Double

Entidade: Servico	
Id	Int
Faturald	Int

Atributo Multivalor: criança	
Id	Int
Nome	String
Idade	Int

4.1.6 Identificação de Chaves Primárias

De maneira a identificar unicamente a ocorrência das entidades foi necessário determinar chaves primárias.

Nenhuma das entidades continha um atributo que identificasse inequivocamente um registo. Como tal, foi necessária atribuir a cada uma destas entidades um atributo **ID**. Este atributo consiste num identificador numérico que identifica um registo de forma única, satisfazendo assim a noção de chave primária.

Assim sendo, não existem chaves candidatas nem alternativas.

4.2. Modelo Lógico

4.2.1 Diagrama do Modelo Lógico

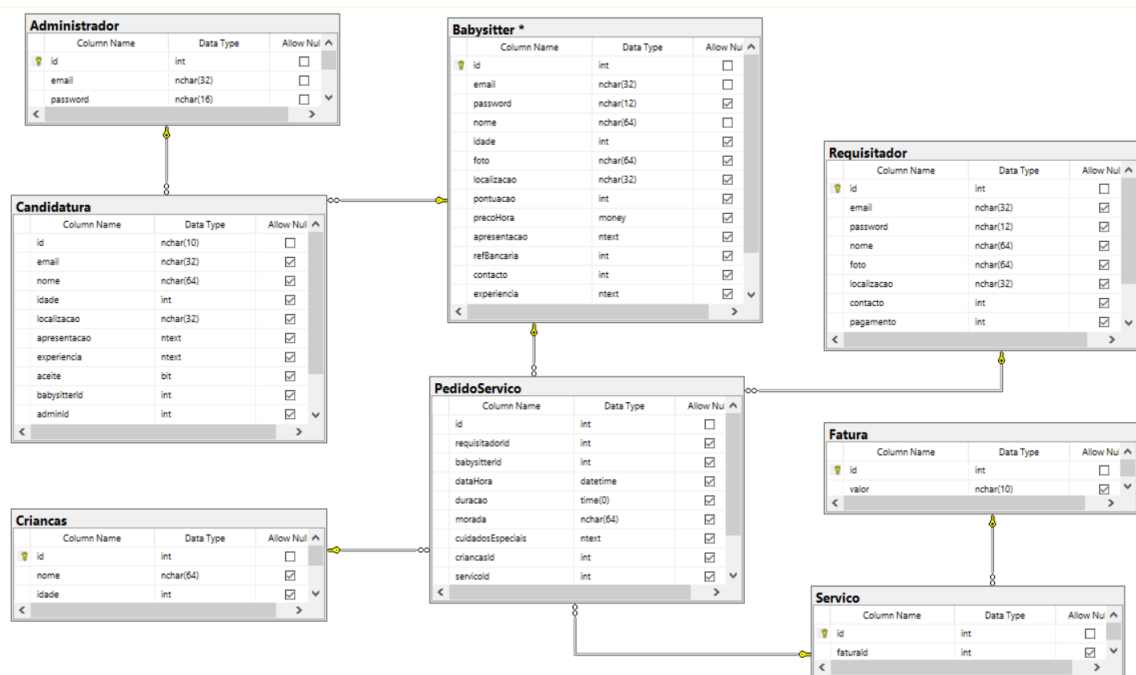


Figura 46 - Modelo Lógico da Base de Dados

4.2.2 Derivação de Relacionamentos

- **Entidades Fortes**

Uma entidade forte é caracterizada pelo facto de possuir uma chave primária que a consiga identificar e distinguir, e que seja independente de outras chaves.

Requisitador (id, email, password, nome, foto, localizacao, contacto, pagamento)

Chave-primária: id

Administrador (id, email, password)

Chave-primária: id

Fatura (id, valor)

Chave-primária: id

Crianças (id, nome, idade)

Chave-primária: id

- **Entidades Fracas**

A existência da entidade fraca depende exclusivamente de outras entidades e serve como relação entre entidades.

PedidoServiço (id, requisitadorId, babysitterId, dataHora, duracao, morada, cuidadosEspeciais, criancasId, servicoId)

Chave-primária: id

Chave-estrangeira: requisitadorId, babysitterId, criancasId, servicoId

4.2.3 Validação Através da Validação

Após analisarmos as dependências funcionais de cada relação verificamos que as tabelas respeitam as três primeiras regras de normalização e, portanto, encontram-se normalizadas até à Terceira Forma Normal da Normalização.

4.2.4 Viabilidade de Crescimento

A longevidade de um sistema de bases de dados depende, sobretudo, da sua capacidade em se adaptar a novos requisitos funcionais que surjam. Situações onde um Modelo Lógico só consegue suportar os requisitos para o qual foi projetado podem tornar o sistema rapidamente obsoleto ou fazer com que o custo de implementação das alterações necessárias seja demasiado dispendioso.

O modelo apresentado está limitado ao negócio do *babysitting*, no entanto, é possível imaginar outras funcionalidades que seriam implementadas na base de dados de forma a que a

sua utilidade melhorasse. Por exemplo, implementar também a aplicação mobile para os requisitadores, ou de forma a que abranja outros serviços, aplicação iOS, por exemplo.

5. *Mockups* da Interface

De seguida são apresentados os mockups relativos à interface do nosso sistema.

Os mockups foram separados em duas partes: interface mobile e interface web. Como visto anteriormente, a aplicação disponível no smartphone irá servir como auxílio às babysitters para facilitar, tanto a marcação e aceitação dos serviços pretendidos pelo utilizador, como também em organizar e relembrar as tarefas necessárias para cada contratação. No que toca ao website, este irá ser utilizado pelos utilizadores que pretendem contratar os nossos serviços.

Com estes mockups, pretendemos mostrar um esboço da interface que iremos dispor aos nossos utilizadores e babysitters.

Primeiramente iremos mostrar as ilustrações relativas à interface mobile e logo a seguir as relativas à interface web.

5.1. Login – mobile

Tendo entrado com sucesso na sua conta, irá-se deparar com a página inicial onde tem acesso às diferentes funcionalidades da aplicação.



Figura 47 - *Mockup* do ecrã de login (aplicação móvel)

5.2. Home – mobile

Tendo entrado com sucesso na sua conta, irá-se deparar com a página inicial onde tem acesso às diferentes funcionalidades da aplicação.



Figura 48 - Mockup do ecrã de home (aplicação móvel)

5.3. Lista de serviços – mobile

No que toca à interface, a lista de serviços e os serviços pendentes têm a mesma disposição pelo que são representados num único *mockup*

Nesta parte, são mostrados os diferentes serviços que a babysitter tem pendentes ou agendados.



Figura 49 - *Mockup* da lista de serviços (aplicação móvel)

5.4. Serviço - mobile

Ao clicar num serviço da lista de serviços marcados, irá aparecer este menu com as diferentes opções relativas ao serviço agendado.

Aqui poderá ter acesso ao mapa onde se encontra sinalizada a localização da morada à qual se deve dirigir, a lista das tarefas solicitadas pelo utilizador, ao contacto facilitado com o utilizador caso necessário, e também um botão para finalizar o serviço como realizado.



Figura 50 - *Mockup* do ecrã de serviço (aplicação móvel)

5.5. Aceitação de serviços – mobile

Ao seleccionar um serviço que se encontra pendente, existe a opção de aceitar ou recusar caso não seja possível realizá-lo. Nesta mesma janela, temos ainda a disponibilidade de ver as informações relativas ao contrato como a morada, hora, tarefas, ...

É ainda possível escrever um pequeno texto, caso necessite, para relatar alguma informação relevante ao serviço.



Figura 51 - *Mockup* de aceitação de serviços (aplicação móvel)

5.6. Login – web

Tal como é feito na aplicação *mobile*, o utilizador necessita também de iniciar sessão com as suas credenciais. Nesta interface *web*, o utilizador tem agora acesso à página de registo onde pode criar uma conta.

Também é disponível um *link* de acesso à página onde é possível se candidatar para babysitter. A sua candidatura é submetida e revista por moderadores para aprovação.

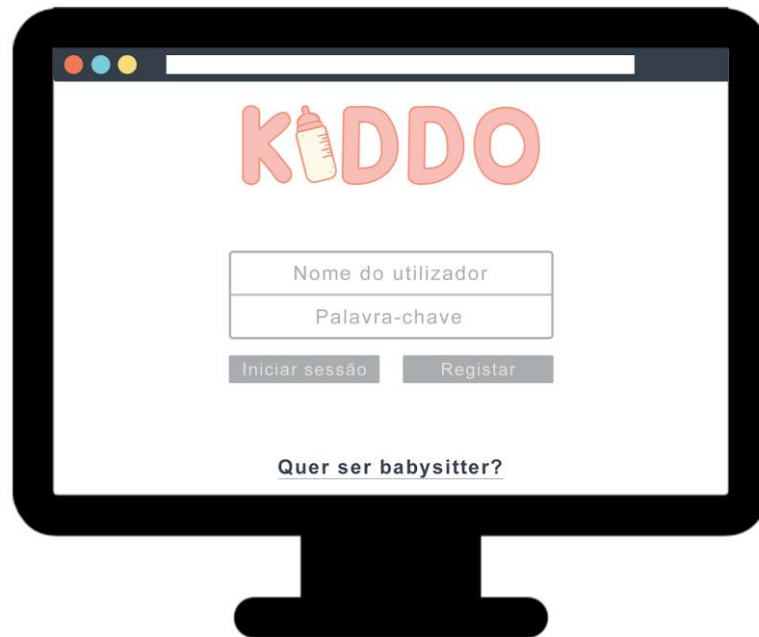


Figura 52 - Mockup do ecrã de login (web)

5.7. Home – web

Efetuada o *login*, o utilizador é apresentado com a página inicial onde pode aceder às várias opções disponíveis.



Figura 53 - *Mockup* do ecrã de home (web)

5.8. Requisitar serviço – web

Ao seleccionar a opção “requisitar serviço”, o utilizador é disponibilizado com duas listas com inúmeras escolhas de babysitters. Uma das listas é relativa às nossas sugestões, tendo em conta a localização, avaliação entre outros. A segunda lista é relativa às babysitters das quais se encontram na sua lista de favoritos ou que recentemente contratou para um serviço.

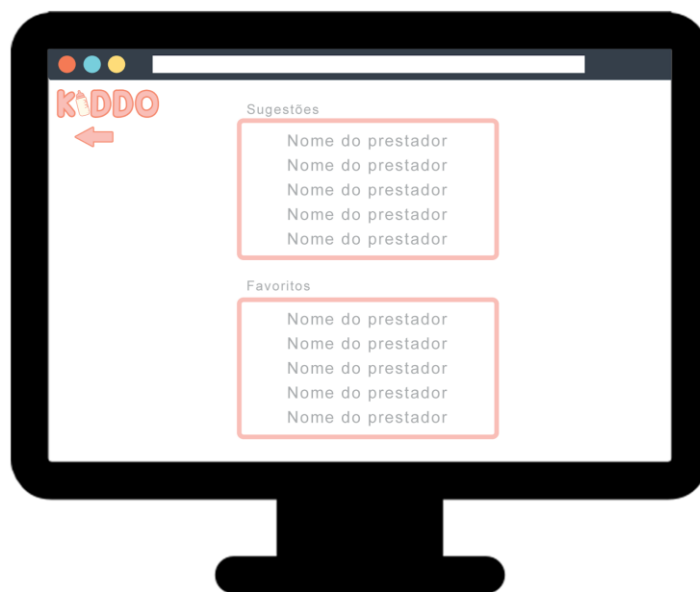
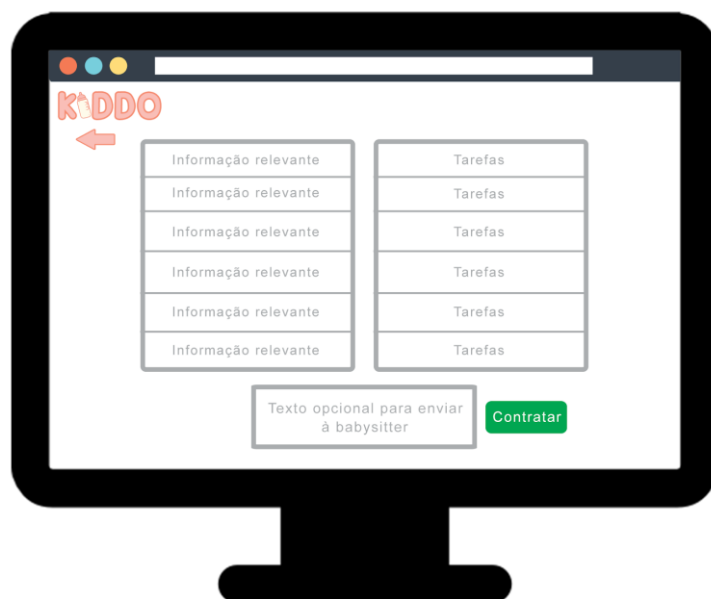


Figura 54 - Mockup do ecrã de requisitar serviço (web)

5.9. Contratar – web

Depois de seleccionada a babysitter que pretende contratar, e-lhe fornecido alguma informação relevante à mesma, bem como uma lista em que pode escrever uma série de tarefas necessárias a realizar no serviço.

Pode ainda depois escrever um pequeno texto em que pode fornecer algumas informações necessárias ao bom funcionamento do serviço.



The mockup shows a web browser window with the Kuddo logo in the top left corner. The main content area contains two side-by-side tables. The left table has six rows, each labeled 'Informação relevante'. The right table has six rows, each labeled 'Tarefas'. Below these tables is a text input field with the placeholder text 'Texto opcional para enviar à babysitter'. To the right of this field is a green button labeled 'Contratar'.

Figura 55 - Mockup do ecrã de fazer pedido de serviço (web)

6. Conclusões e Trabalho Futuro

Após o término desta segunda fase damos por concluída a especificação do nosso projeto. Conseguimos, favoravelmente, fazer o levantamento dos requisitos do utilizador e, tendo-os como base, estruturar requisitos do sistema concisos e robustos, tanto funcionais como não funcionais.

Incorporou-se igualmente alguns diagramas UML que considerámos relevantes para uma melhor modelação e compreensão do sistema que irá ser desenvolvido. Por exemplo, o diagrama de Use Cases proporcionou-nos uma noção mais visual de como irão ocorrer as interações utilizador-sistema e modelo de domínio permitiu-nos avaliar mais adequadamente quais os elementos que vão constituir o sistema.

De seguida, iniciámos a execução dos diagramas, a idealização e estruturação da base de dados. Realizou-se o modelo conceptual, validando-a com o cliente, e estabeleceu-se então o modelo lógico. Visando um bom desempenho e uma redução na redundância de dados aplicaram-se as três formas normais e garantiu-se um modelo normalizado da nossa base de dados.

Além de tudo isto, debruçámo-nos sobre o esboço de mockups, idealizando a interface e camadas de apresentação dos dois sistemas.

Para a última fase deste projeto, pretendemos atingir uma implementação sólida, baseado no trabalho desenvolvido até ao momento, num aperfeiçoamento gradual até a um produto final com que esperamos satisfazer as expectativas do cliente.

7. Referências

Sommerville, I. (2011). Software engineering. 9th ed. Boston, Massachusetts: Addison-Wesley

Bootstrap 4 – <https://getbootstrap.com/>

Google Maps – <https://developers.google.com/maps/>