3ºAno

Desenvolvimento de Sistemas de Software

Licenciatura Engenharia Informática

Janeiro 2024

GRUPO 23

Pedro Miguel da Costa Azevedo A100557

Miguel Tomás Antunes Pinto A100815

Ana Margarida Sousa Pimenta A100830

Inês Gonzalez Perdigão Marques A100606

Introdução

Este projeto propõe a criação e implementação de um sistema automatizado para agilizar o processo de admissão e encaminhamento de clientes na *E.S.Ideal*.

O objetivo deste trabalho é então otimizar a experiência do cliente melhorando a eficiência operacional diante do crescente influxo de clientes.

Deste modo, seguindo os métodos propostos nas aulas teóricas e práticas de *UC* de *Desenvolvimento de Sistemas de Software*, começamos por efetuar o respetivo levantamento de requisitos.

Quanto à progressão no projeto, iremos evidenciar através dos diagramas e métodos desenvolvidos que serão exemplificados neste relatório.

Método Adotado

Inicialmente, começamos por identificar as entidades do problema e elaboramos o *Diagramas de Domínio* para compreender de forma mais eficaz e direta o contexto do problema. De seguida, passamos à modulação do mesmo usando *Diagramas de Use Cases* para visualizar interações usuário-sistema. Depois, tendo em conta a lógica de negócio adotada decidimos subdividir o nosso sistema principal, a oficina, em subsistemas, que se encontram representados no *Diagrama de Componentes*.

Seguidamente, a arquitetura de cada subsistema foi definida por meio de *Diagramas de Classes* e, para validar as operações, elaboramos *Diagramas de Sequência*, que descrevem a interação entre objetos.

Mais adiante na implementação, vamos também ajustar a arquitetura para incluir *DAOs* em vez de *maps*, visando à persistência de dados numa base de dados.

Quanto a este último ponto, queremos sobretudo enfatizar a importância dos *DAOs* para a persistência dos dados, pois este método concentra a lógica de acesso aos dados num único local.

1. Modelo de Domínio

De modo a compreender melhor o problema que nos foi proposto decidimos começar por fazer o digrama de domínio onde mostramos a entidades e atributos que definimos.

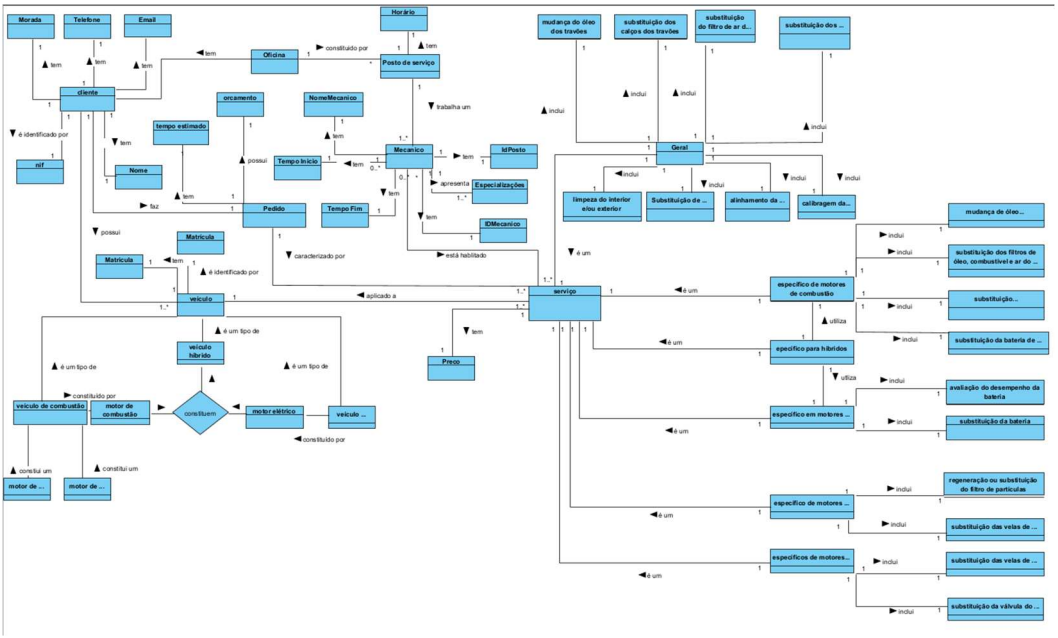

Figura 1 – Modelo de Domínio

2. Diagrama de Use Cases

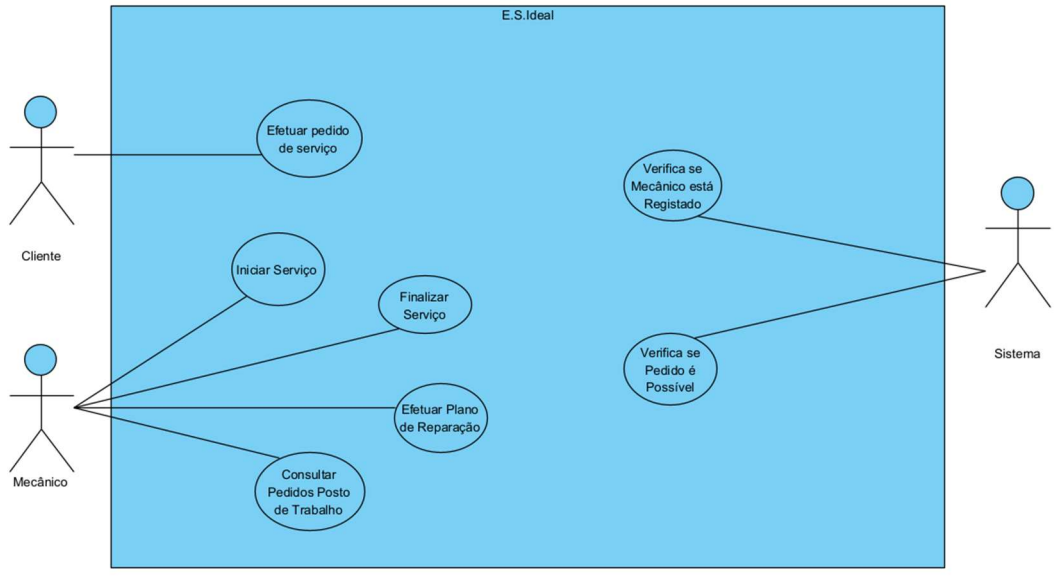

Figura 2 - Diagrama de Use Cases

Use Cases

Use Case: *Efetuar Pedido de Serviço*

Descrição: Cliente efetua um pedido de serviço para o seu veículo.

Ator: Cliente da E.S.Ideal

Pré-Condição: O cliente apresenta registo na E.S.Ideal.

Pós-Condição: O serviço pedido fica registado.

Fluxo Normal:

1. O cliente efetua um pedido de serviço.
2. O sistema regista o pedido do cliente.
3. O funcionário realiza o check-up do veículo.

Use Case: *Efetuar plano de reparação*

Descrição: O funcionário efetua um plano de reparação para o veículo.

Ator: Funcionário

Pré-Condição: O funcionário tem de estar registado.

Pós-Condição: O veículo apresenta na sua ficha um plano de reparação.

Fluxo Normal:

Ator

1. O funcionário realiza o check-up do veículo.
2. O veículo necessita de mais reparações.
3. O funcionário cria um plano de reparação para o veículo.
4. O sistema atualiza a ficha do veículo.

Fluxo Alternativo:

2.1. O veículo não precisa de mais reparações.

2.2. O sistema atualiza a fica do veículo.

Use Case: Consultar pedidos de um posto de trabalho

Descrição: O funcionário pode verificar a lista de pedidos associados a um determinado posto de trabalho.

Ator: Mecânico

Pré-Condição: O mecânico está registado (existe) e pertence ao posto.

Pós-Condição: O mecânico visualiza a lista de pedidos para o respetivo posto de trabalho.

Fluxo Normal:

1. O mecânico solicita a consulta da lista de pedidos associados a um posto de trabalho específico.

2. O sistema verifica a existência do posto de trabalho com base no identificador fornecido.

3. Se o posto de trabalho existe, o sistema recupera a lista de pedidos associados a esse posto.

3. O sistema exibe a lista de pedidos para o mecânico, incluindo informações relevantes como número do pedido, cliente e veículo, por exemplo.

Fluxo Alternativo:

2.1 O sistema verifica que o posto de trabalho não existe.
2.2 O sistema informa o mecânico que o posto de trabalho não existe.

Use Case: Verifica se mecânico está registado

Descrição: O sistema verifica se um mecânico está registado.

Ator: Sistema

Pré-Condição: O mecânico fornece a sua identificação ao sistema.

Pós-Condição: O sistema indica se o mecânico está registado ou não.

Fluxo Normal:

1. O sistema recebe a solicitação para verificar se um mecânico está registado.

2. O sistema verifica as informações fornecidas pelo mecânico.

3. Se as informações forem válidas, o sistema verifica se o mecânico está registado no sistema.

3. O sistema indica se o mecânico está ou não registado.

Fluxo Alternativo:

3.1 O sistema determina que o mecânico não está registado.
3.2 O sistema informa que o mecânico não existe.

Use Case: Inicializar pedido

Descrição: O mecânico inicia um pedido.

Ator: Mecânico

Pré-Condição: O mecânico está registado (existe) e pertence ao posto respetivo ao pedido.

Pós-Condição: O pedido em causa é dado como iniciado.

Fluxo Normal:

1. O mecânico solicita a inicialização de um pedido.

2. O sistema verifica se o mecânico tem permissão para inicializar pedidos.

3. O sistema verifica se o pedido existe e está pronto a ser iniciado.

4. Se todas as verificações forem bem-sucedidas, o sistema inicia o pedido e atualiza o seu estado.

5. O sistema indica que o pedido foi iniciado com sucesso.

Fluxo Alternativo:

3.1 O sistema verifica que o pedido não existe.

3.2 O sistema indica que o pedido em causa não existe.

Use Case: Finalizar pedido

Descrição: O mecânico finaliza um pedido.

Ator: Mecânico

Pré-Condição: O mecânico está registado (existe) e pertence ao posto respetivo ao pedido.

Pós-Condição: O pedido em causa é dado como finalizado.

Fluxo Normal:

1. O mecânico solicita a finalização de um pedido.

2. O sistema verifica se o mecânico tem permissão para finalizar pedidos.

3. O sistema verifica se o pedido existe e está pronto a ser finalizado e, se foi previamente iniciado.

4. Se todas as verificações forem bem-sucedidas, o sistema finaliza o pedido, atualizando a sua informação.

5. O sistema indica que o pedido foi finalizado com sucesso.

Fluxo Alternativo:

3.1 O sistema verifica que o pedido não existe.

3.2 O sistema indica que o pedido em causa não existe.

Use Case: Verificar se pedido é possível

Descrição: O sistema verifica se um pedido é possível verificando se o cliente, o veículo e o serviço existem.

Ator: Sistema

Pré-Condição: O cliente fornece informações válidas (cliente, veículo, serviço) para criar um pedido no sistema.

Pós-Condição: O sistema indica se o pedido é possível tendo em conta a informação inserida pelo cliente.

Fluxo Normal:

1. O cliente solicita a criação de um pedido.
2. O sistema valida as informações fornecidas pelo cliente (cliente, veículo, serviço).
3. O sistema verifica a existência do cliente no sistema.
4. O sistema verifica a existência do veículo no sistema.
5. O sistema verifica a existência do serviço no sistema.
6. O sistema confirma que o pedido é possível.
7. O pedido é registado com sucesso.

Fluxo Alternativo (Cliente não existe):

3.1 O sistema determina que o cliente não existe.

3.2 O sistema indica que o cliente não existe.

Fluxo Alternativo (Veículo não existe):

4.1 O sistema determina que o veículo não existe.

4.2 O sistema indica ao cliente que o veículo não existe.

Fluxo Alternativo (Serviço não existe):

5.1 O sistema determina que o serviço não existe.

5.2 O sistema indica ao cliente que o serviço inserido não existe.

Fluxo Exceção (Pedido não é possível):

6.1 O sistema determina que o pedido não é possível.

6.2 O pedido não é concluído com sucesso.

3. Diagrama de Componentes

O diagrama de componentes é uma representação visual que mostra as partes principais de um sistema e como essas partes estão conectadas. Este é usado na modelação de sistemas de modo a ilustrar a estrutura e as dependências entre os componentes de um sistema.

Abaixo, exemplificamos como está representado o nosso sistema:

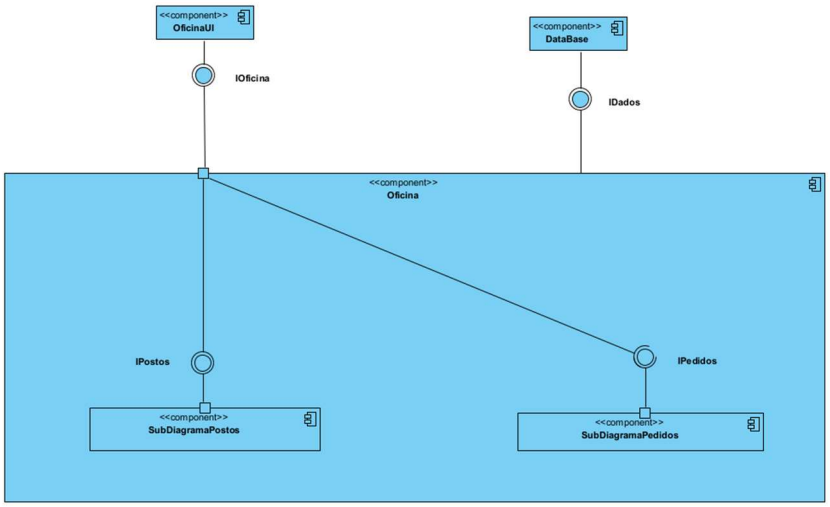

Figura 3 - Diagrama de Componentes

4. Diagrama de Classes

De seguida, apresentamos o nosso diagrama de classes.

Para esta fase decidimos dividir o nosso sistema Oficina em dois subsistemas, sendo um para os Postos e outro para os Pedidos.

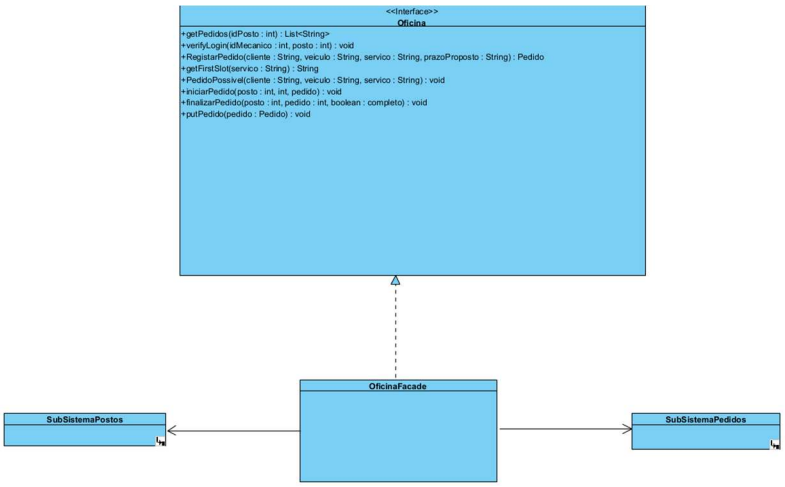

Figura 4 - Diagrama de Classes

Quanto aos subdiagramas, começamos então pelo diagrama respetivo aos Pedidos:

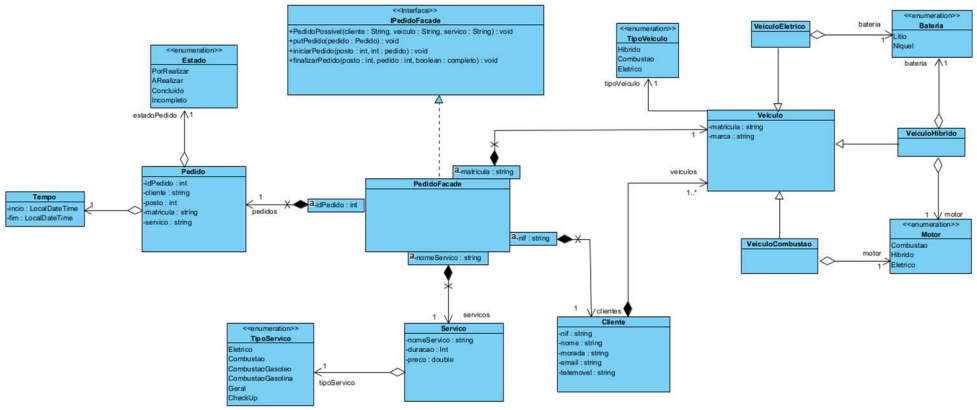

Figura 5 - Subdiagrama Pedido

Temos agora o subdiagrama dos Postos:

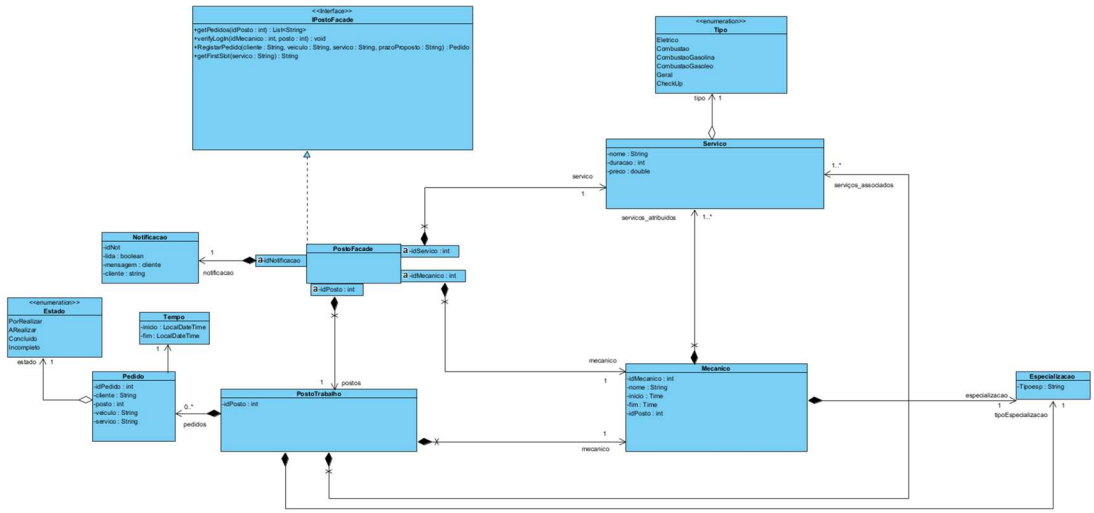

Figura 6 - Subdiagrama dos Postos

5. Diagramas de Sequência

Agora iremos representar os diagramas de sequência.

Estes diagramas são importantes para compreender a dinâmica entre as variadas componentes deste projeto ao longo tempo do seu processamento.

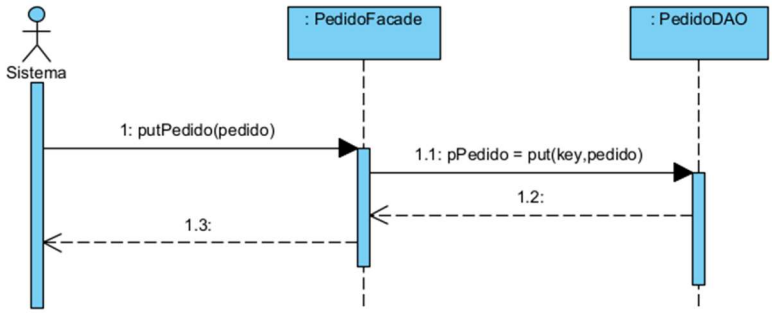

Figura 7 - Diagrama de Sequência putPedido

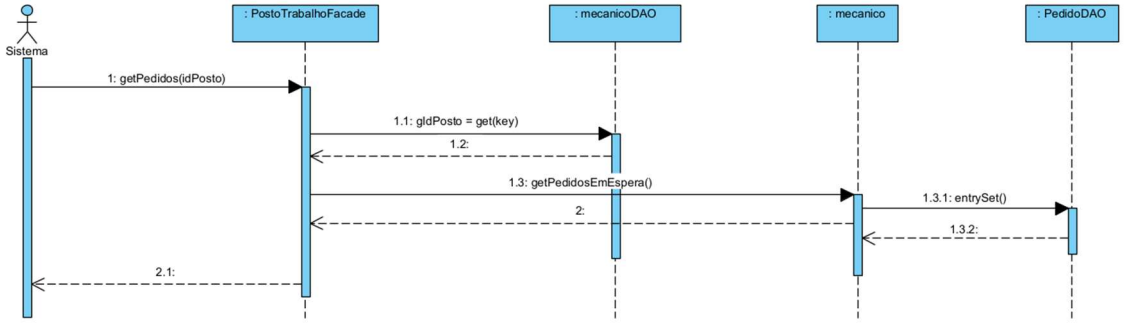

Figura 8 - Diagrama de Sequência getPedido

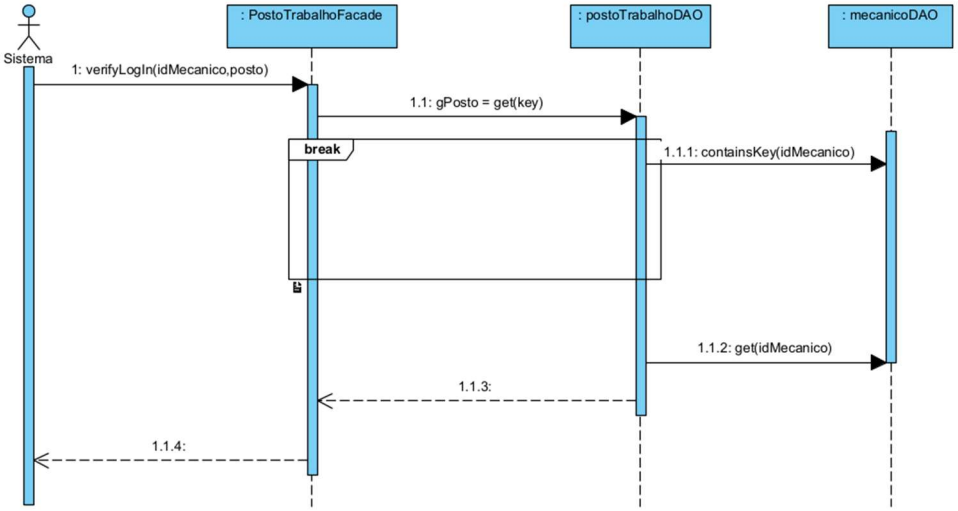

Figura 9 - Diagrama de Sequência da verifyLogIn

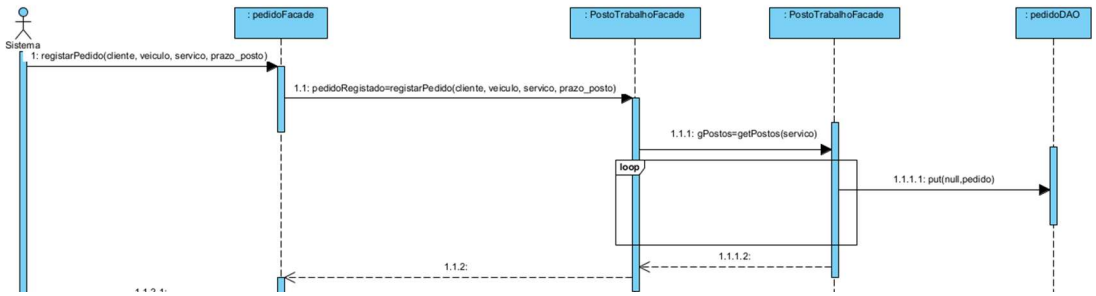

Figura 10 - Diagrama de Sequência da registarPedido

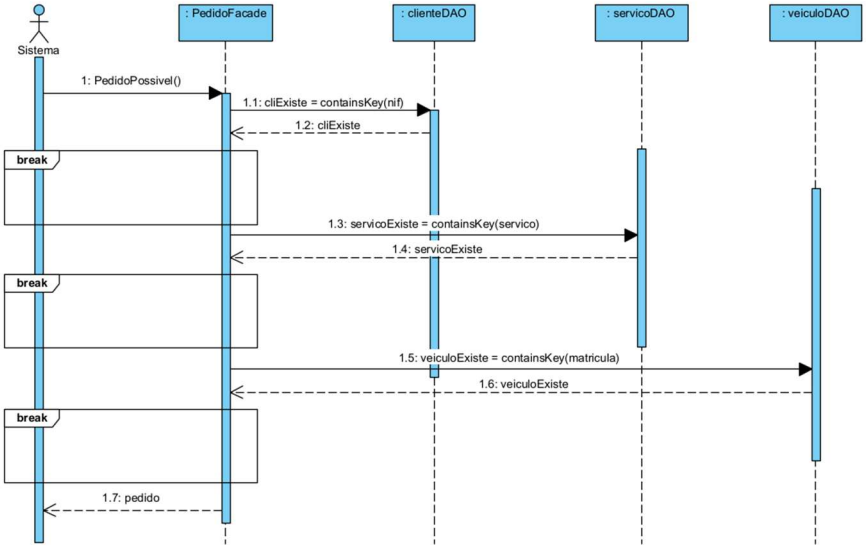

Figura 11 - Diagrama de Sequência da PedidoPossivel

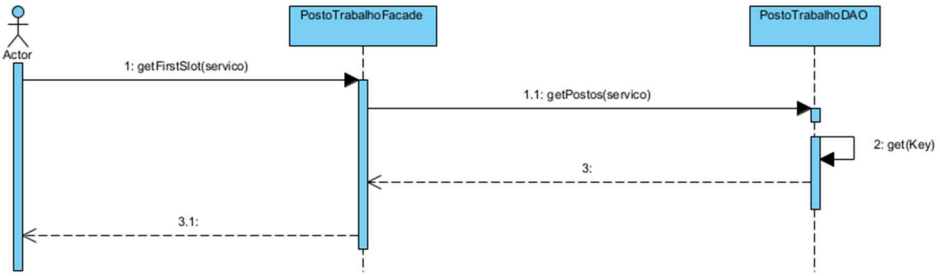

Figura 12 - Diagrama de Sequência da getFirstSlot

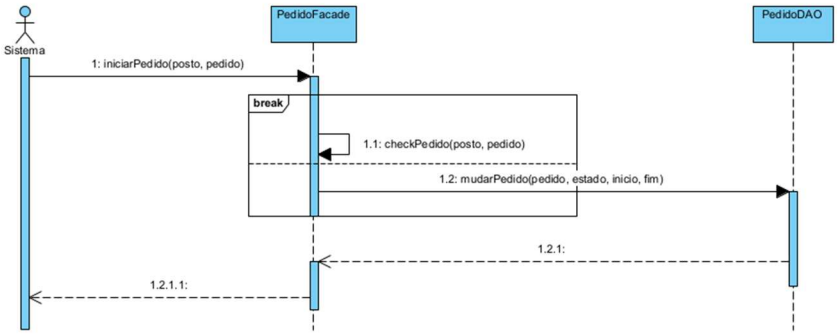

Figura 13 - Diagrama de Sequência da iniciarPedido

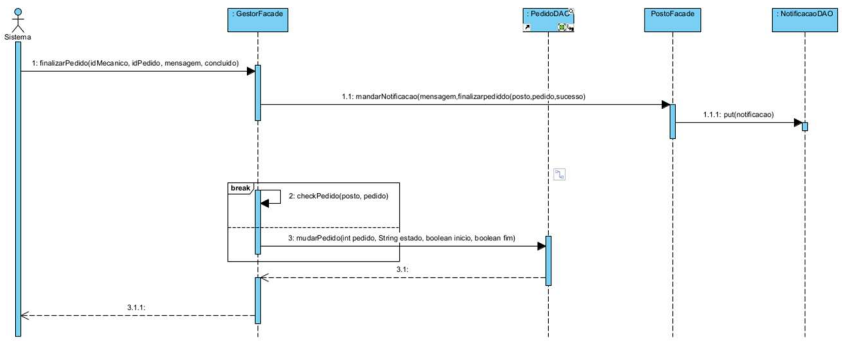

Figura 14 - Diagrama de Sequência da finalizarPedido

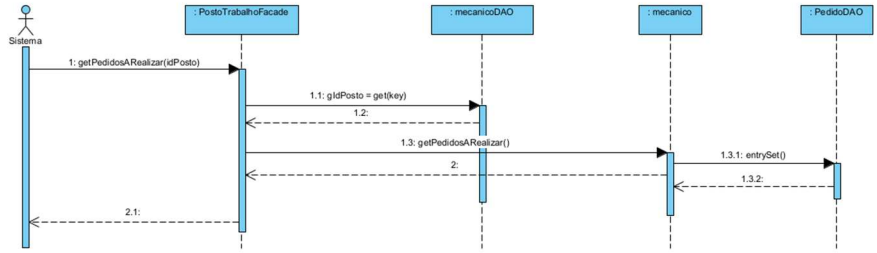

Figura 15 - Diagrama de Sequência da getPedidoArealizar

6. Diagramas com DAOS

Aos subdiagramas abordados previamente, decidimos implementar os *DAOS*.

Consideramos que esta é uma implementação importante uma vez que esta é responsável por permitir o acesso à base de dados.

Temos aqui o subdiagrama dos Postos, agora implementado com os *DAOS*:

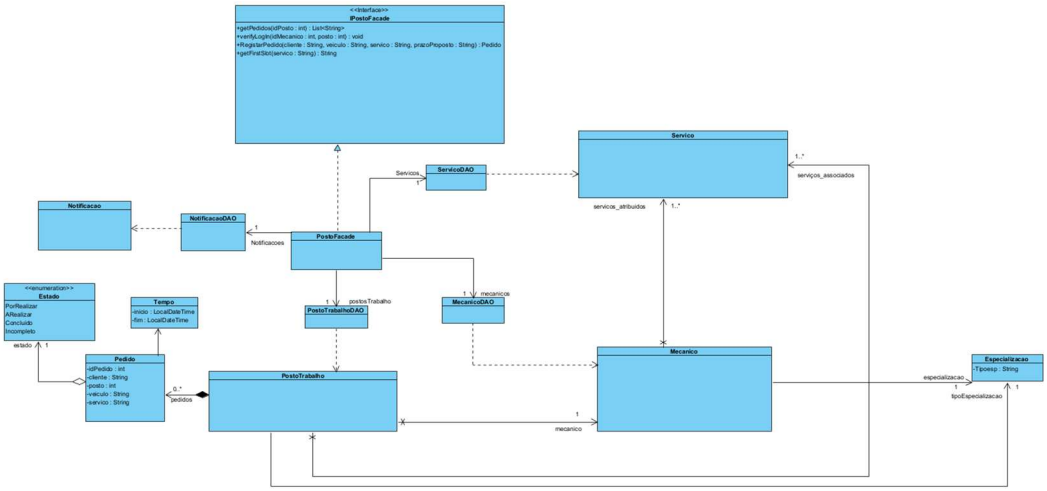

Figura 16 - Subdiagrama Postos com DAOS

Quanto ao subdiagrama dos pedidos, obtivemos este diagrama:

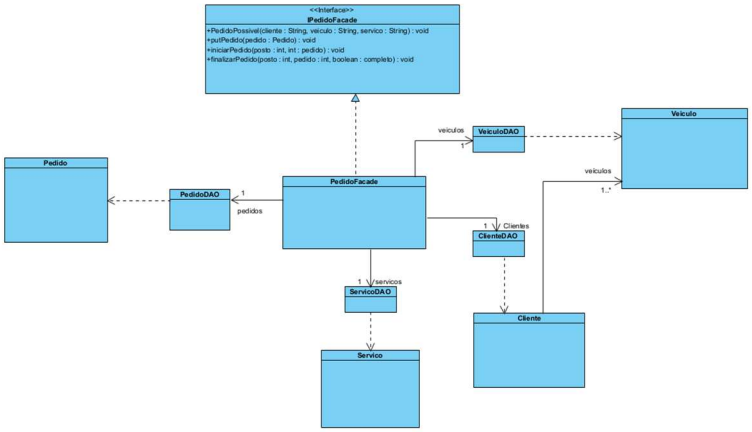

Figura 17 - Subdiagrama dos Pedidos com DAOS

7. *Diagrama de Pacotes*

Um diagrama de pacotes é um diagrama de estrutura que permite verificar como estão organizados (em pacotes) os elementos de um sistema.

Aqui, apresentamos o nosso diagrama de pacotes:

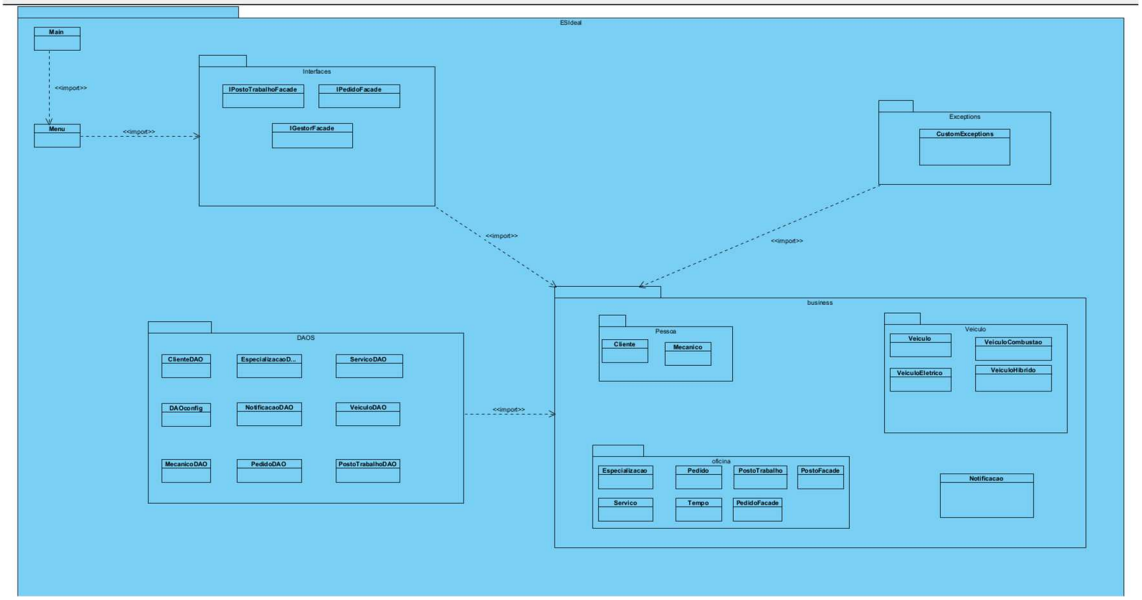

Figura 18 - Diagrama de Pacotes

Conclusão e Resultados Obtidos

Em suma, sobre este projeto, refletimos na sua importância e experiência que este nos proporcionou.

Durante a realização deste projeto enfrentamos alguns problemas e desafios, surgiram alguns erros na implementação do código o que levou a posteriores dúvidas e alterações nos respetivos diagramas. No entanto, consideramos que todos esses problemas foram resolvidos e esclarecidos com os docentes.

Apesar dos desafios enfrentados, estamos satisfeitos com o resultado do projeto, considerando-o concluído de maneira satisfatória. Reconhecemos, no entanto, que mais tempo nos permitiria aprimorar ainda mais o trabalho, tornando-o mais completo e apresentável.