Universidade do Minho
Escola de Engenharia

Desenvolvimento de Sistemas de Software

Grupo 1

João Pedro da Santa Guedes A89588
Luís Pedro Oliveira de Castro Vieira A89601
Carlos Miguel Luzia Carvalho A89605
Bárbara Ferreira Teixeira A89610


A89588


A89601


A89605


A89610

23 de dezembro de 2020

Índice

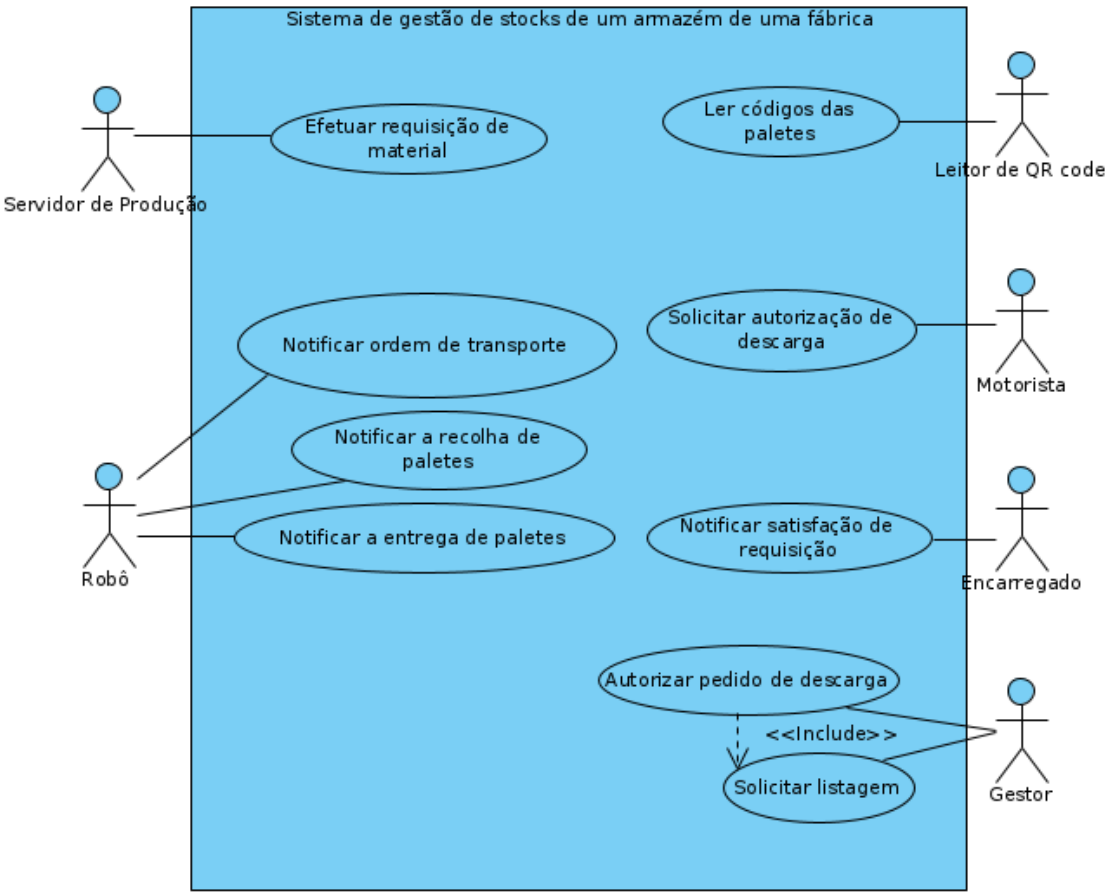
1 Introdução

Na terceira fase do trabalho tivemos como obejtivo implementar o projeto em java, utilizando uma database com o objectivo de suportar o processo de gestão das paletes, desde que o código QR é lido, até que são armazenadas. Os Use Case a considerar para cada actor são: **Solicitar Listagem, Ler QR Code, Notificar a necessidade de transporte de paletes para entregas, Notificar a necessidade de transporte de paletes para armazenamento, Notificar sucesso de entregar**, tendo estes sido ligeiramente alterados devido ao nosso maior entendimento do projeto, e ao detetar problemas nos submetidos em fases anteriores.

2 Use Cases

Ao longo do desenvolvimento desta terceira fase fomos nos apercebendo de algumas coisas que fizemos não se encontravam corretas ou não era a maneira mais fácil de implementar, nas fases anteriores. Assim, fizemos algumas alterações aos use cases anteriormente apresentados e diagramas de sequencia.

2.1 Diagrama de Use Cases



2.2 Use Case: Solicitar listagem

Descrição: Gestor solicita listagem ao Sistema
Pré-condição: true
Pós-condição: Gestor obtém a informação do armazém

Fluxo normal:
1.Gestor solicita ao sistema a listagem das paletes
2.Sistema apresenta listagem ao gestor.

2.3 Use Case: Ler QR code

Descrição: Leitor de QR code lê código da palete
Pré-condição: Palete foi descarrega e está na fila de paletes a aguardar leitura
Pós-condição: Código da palete é lido, adicionado à lista de códigos lidos e registado no Sistema.

Fluxo normal:
1. Leitor de QR code lê código;
2. Leitor comunica o código ao Sistema;
3. Sistema adiciona código da palete à lista de códigos lidos;
4. Sistema adiciona informações sobre a palete;

Fluxo de Excepção: [leitor de QR code não consegue ler o código] (passo 1):
1.1 Código de palete não é adicionado à lista de códigos lidos
1.2 Palete é descartada e o Sistema remove o código da lista de paletes a aguardar leitura;

2.4 Use Case: Comunicar ordem de transporte

Descrição: Sistema verifica que a fila de paletes à espera de serem transportadas não está vazia, e solicita o seu transporte notificando um robot

Pré-Condição : Fila (queue) de paletes a necessitar de transporte não está vazia

Pós-Condição : Robot fica notificado

Fluxo Normal:
1.Sistema verifica que existe uma prateleira livre;
2.Sistema altera a disponibilidade da prateleira onde a palete vai ser armazenada para indisponível;
3.Sistema procura Robot para transportar a prateleira;
4.Sistema altera a disponibilidade do Robot para indisponível;
6.Sistema elimina a palete da lista de paletes a aguardar transporte;
7.Sistema notifica o Robot;

Fluxo de Exceção 1 [nenhum robot está disponível] (Passo 3):
3.1 Nenhum Robot se encontra disponível para realizar o transporte;
3.2 Sistema volta a colocar a palete na fila de espera;
3.3 Sistema suspende processo e envia uma notificação a avisar que não existem Robots disponíveis;

Fluxo de Exceção 2 [nenhuma prateleira disponível] (Passo 1):
1.1 Nenhuma prateleira se encontra desocupada;
1.2 Sistema volta a colocar a palete na fila de espera;
1.3 Sistema suspende processo e envia uma notificação a avisar que não existe nenhuma prateleira disponível;

2.5 Use Case: Notificar entrega de Paletes

Descrição: Robot recebe o percurso que terá que percorrer para entregar a palete e envia uma notificação no final do processo
Pré-condição: O Robot notificou o sistema que recolheu as paletes
Pré-condição: O sistema fica notificado que o Robot conseguiu concluir a

entrega das paletes

Fluxo Normal:
1.Sistema cria as rotas que o Robot precisa de percorrer para entregar as paletes;
2.Robot entrega a palete;
3.Sistema atualiza a localização da palete para a prateleira onde foi colocada.
4.Sistema atualiza a prateleira, colocando lá a informação da palete armazenada;
5.Sistema atualiza a disponibilidadee do robot para disponível;
6.Sistema atualiza a localização do robot;
7.Robot notifica o Sistema que concluiu a entrega;

2.6 Use Case: Notificar a recolha de paletes

Descrição: O robot recebe o percurso que irá percorrer para recolher as paletes e notifica o sistema no final do processo
Pré-condição: O robot foi notificado que existem paletes a necessitar de transporte
Pós-condição: O Sistema é notificado que o robot recolheu as paletes

Fluxo Normal:
1.Sistema calcula as rotas que o Robot terá que percorrer para entregar as paletes;
2.Robot recolhe as paletes;
3.Sistema atualiza a localização da palete para ser o Robot;
4.Sistema atualiza as informações do Robot para colocar lá a palete que vai transportar;
5.Robot notifica o sistema que concluiu a recolha;

2.7 Explicação das Alterações

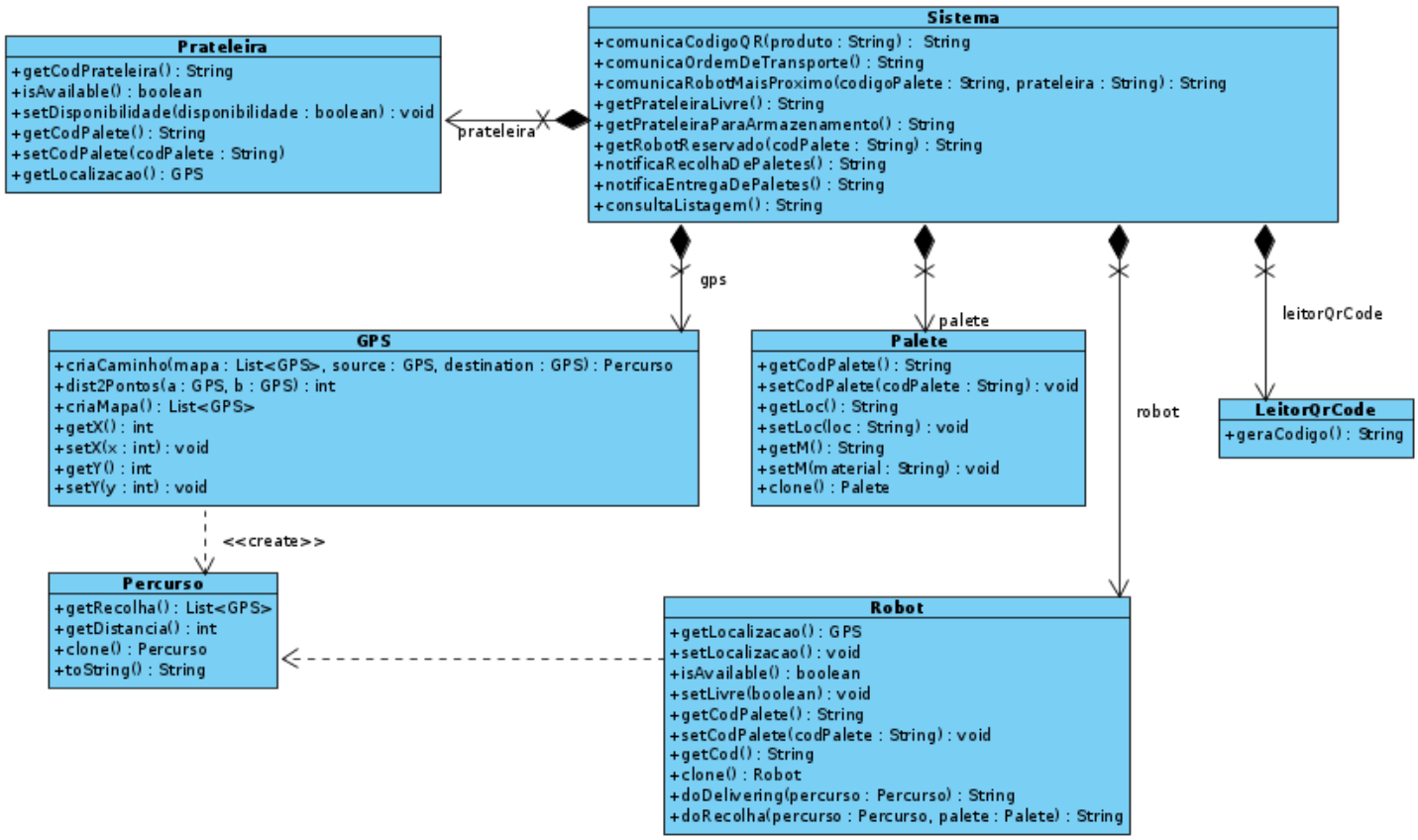
Dada a simplificação do trabalho para esta terceira fase, deixou de ser necessário a diferenciação em vários Use Cases, daí a re dução no número de Use Cases apresentados face à fase previamente entregue. Para além disso, por questões de logística com a Base de Dados implementada, passámos a calcular as rotas que o Robot deve percorrer quer no Use Case de recolha,

quer no Use Case de entrega, uma vez que estas são trabalhadas apenas na memória por serem usadas tão pouco.

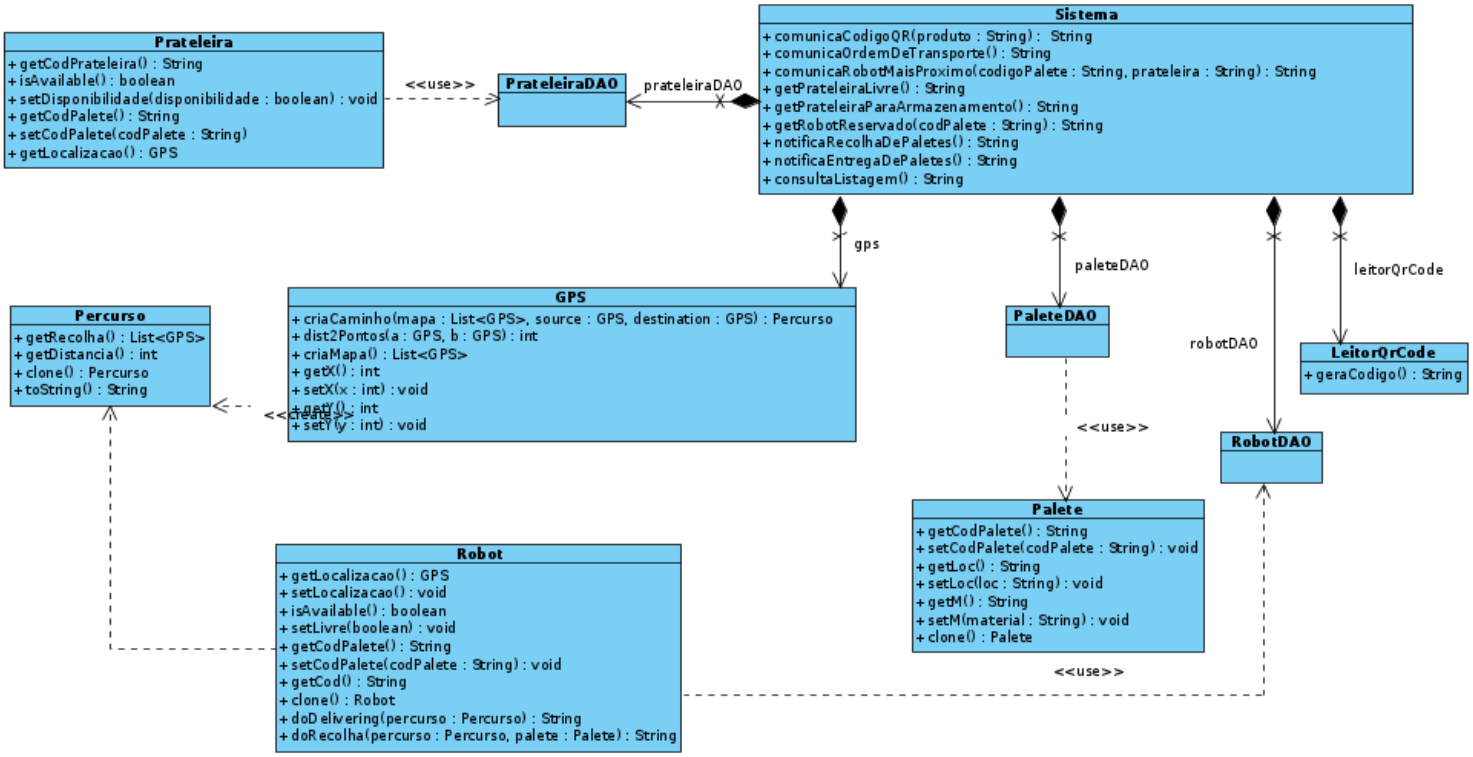
Caso o fizessemos no Use Case de comunicar ordem de transporte, seria necessário armazená-las na base de dados para posteriormente as usarmos, o que seria dificultado pelo facto de se, como veremos mais a frente, classes criadas por nós para o efeito do trabalho, algo díficil de representar na base de dados.

3 Diagrama de Classes

3.1 Diagrama de Classes



3.2 Diagrama de ORM

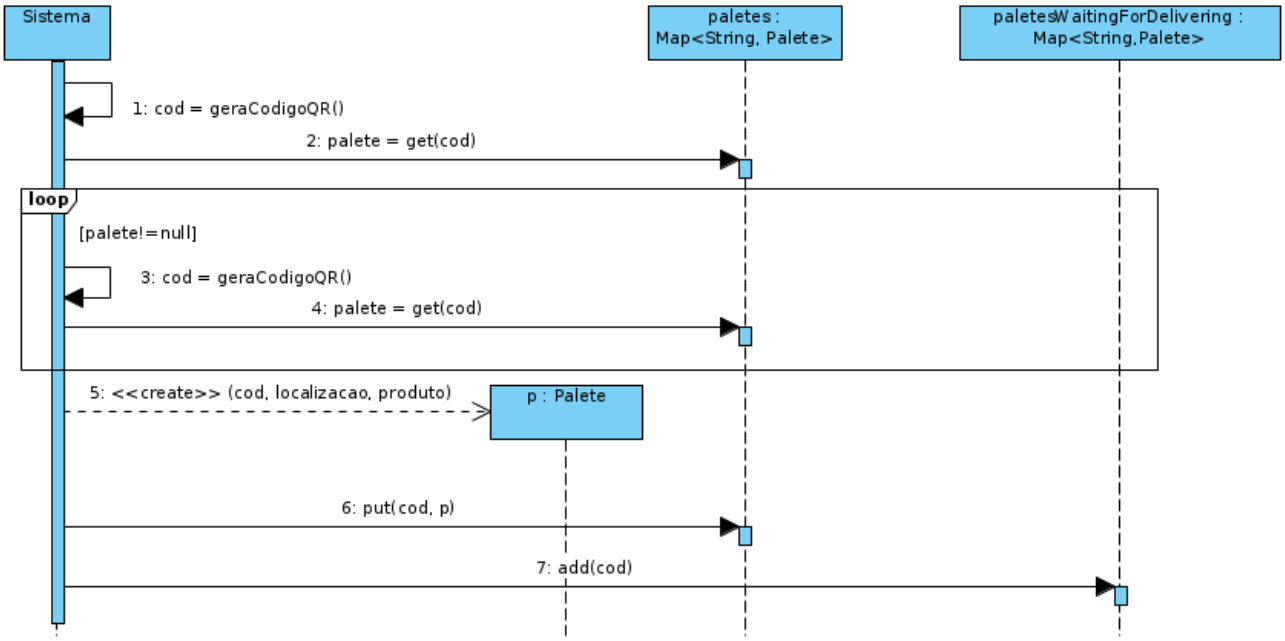


4 Diagramas de Sequência

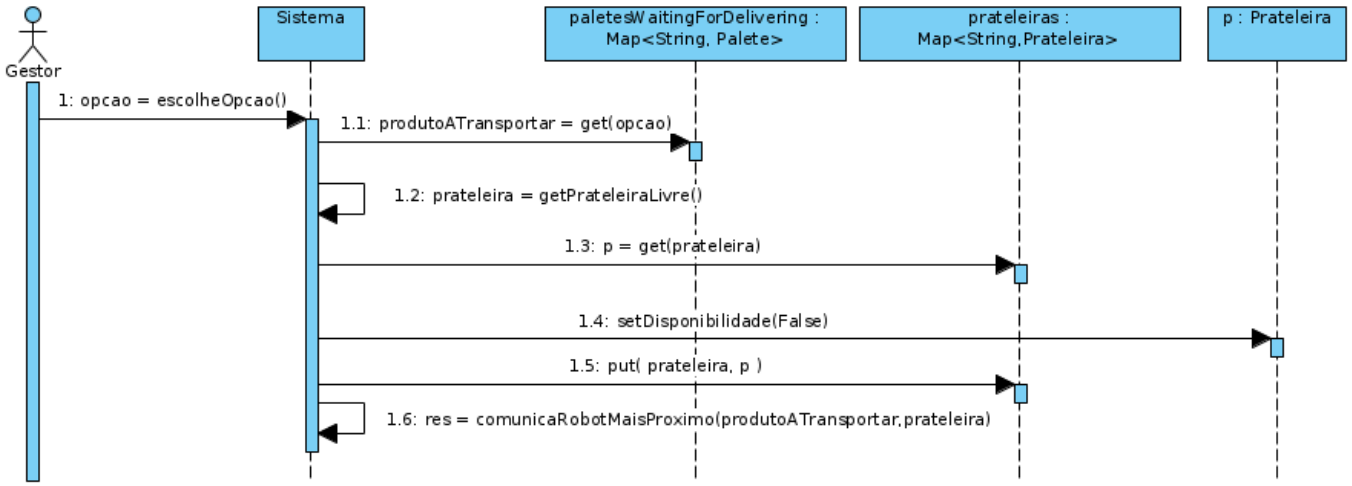
Devido ás alterações efetuadas nos use cases, e à simplificação do trabalho que ocorreu nesta terceira fase, tivemos que alterar alguns dos nossos diagramas de sequência para que estes estivessem de acordo com o nosso trabalho.

Podemos começar por evidenciar que vários diagramas de sequência foram excluídos deste relatório devido a deixarem de ser necessários para o que era pedido nesta fase, como, por exemplo, o login e o logout. Para além disto, compactamos vários diagramas de sequência pelo facto de termos feito o mesmo com certos use cases. Vamos então apresentar os diagramas de sequência com as alterações que achamos necessárias para esta fase.

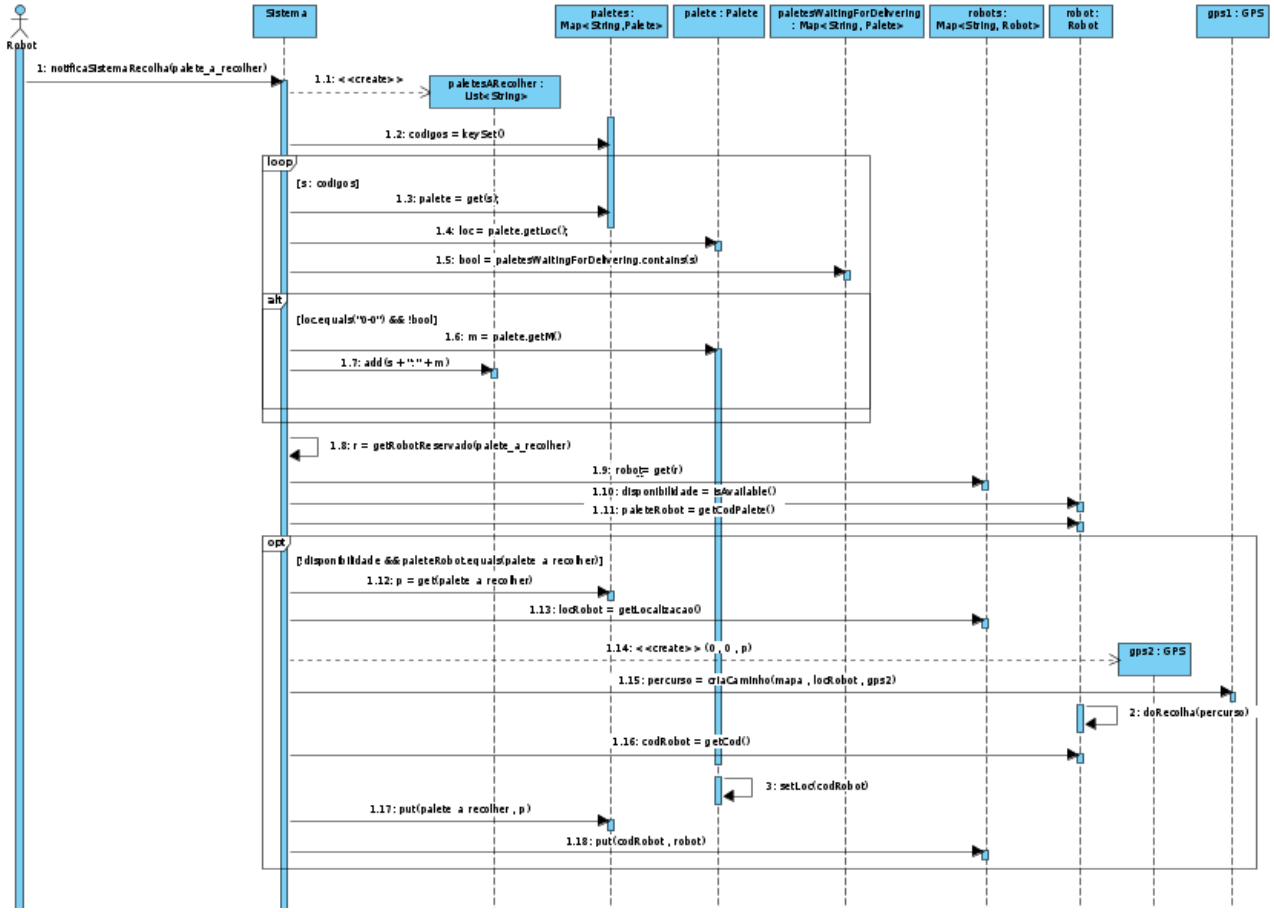
4.1 Ler Código QR



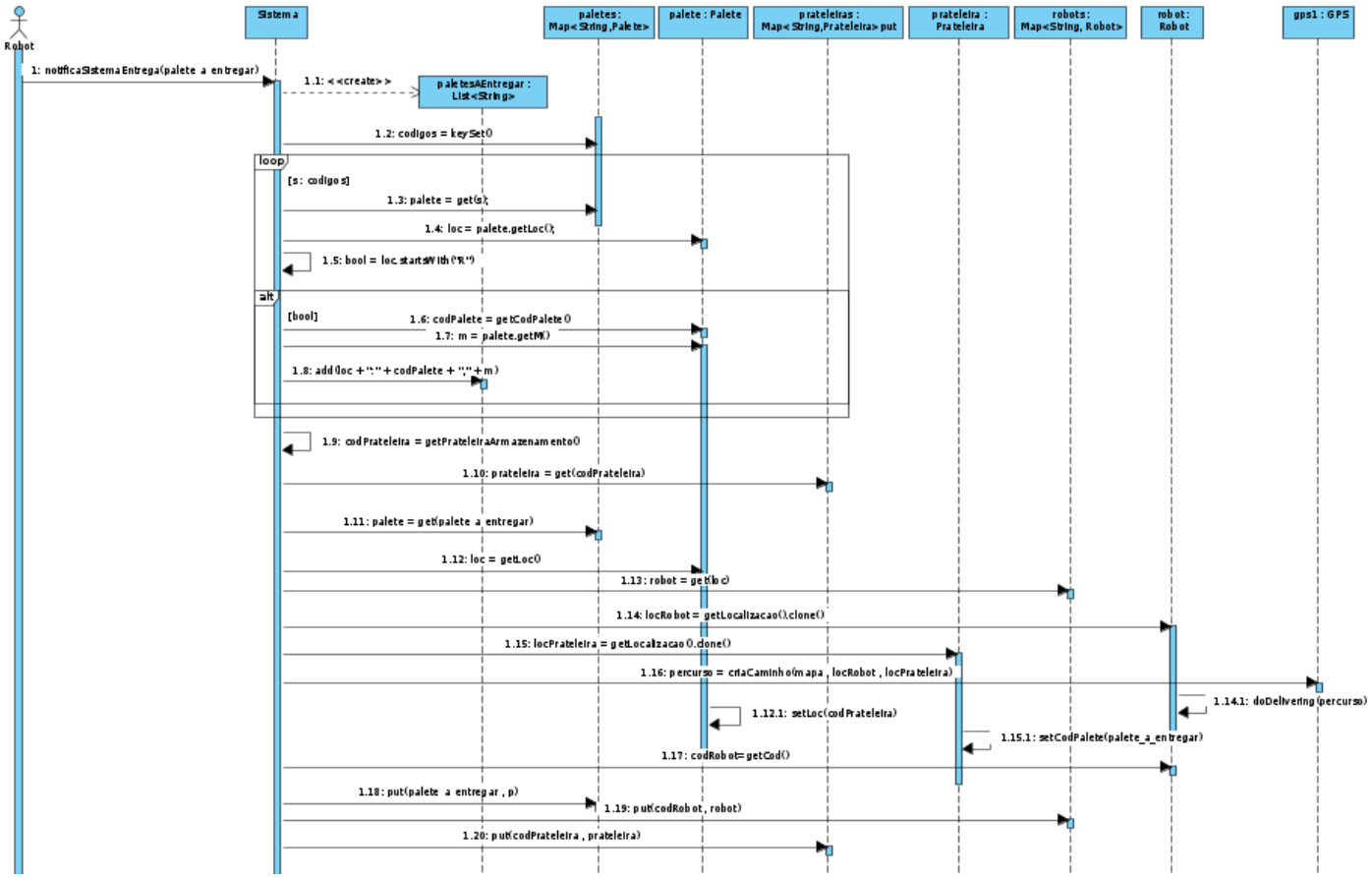
4.2 Notificar a necessidade de transporte de paletes para armazenamento



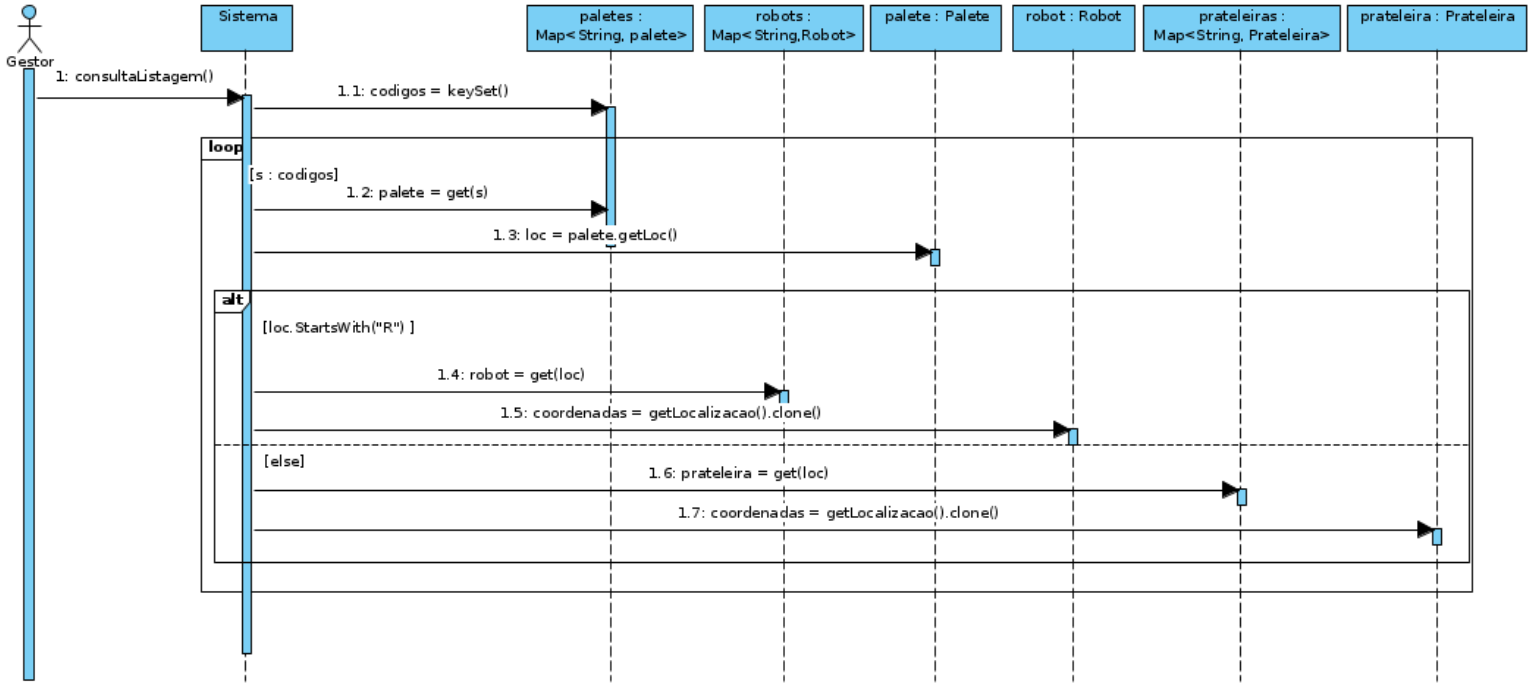
4.3 Notificar o início do transporte das paletes para armazenamento



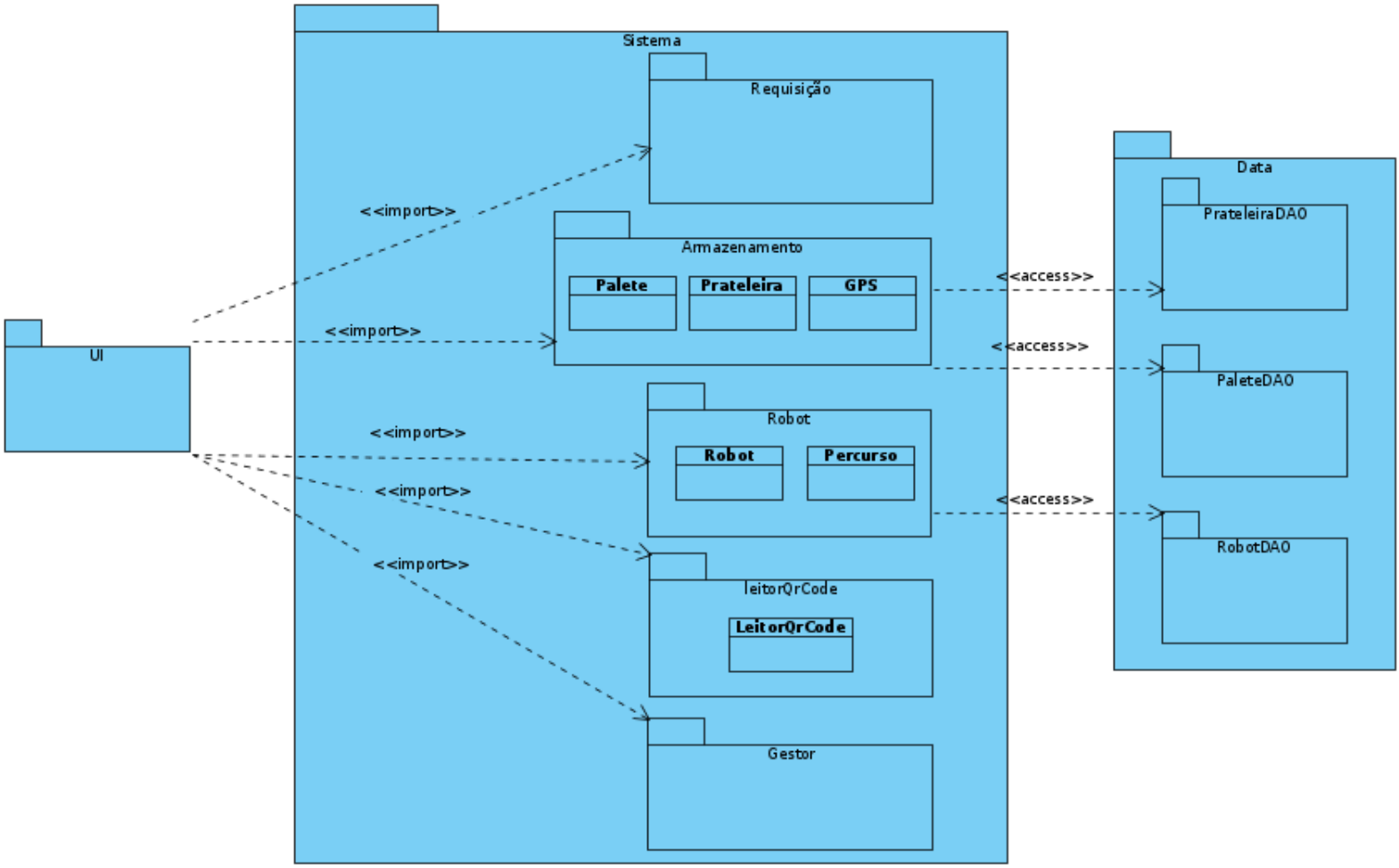
4.4 Notificar sucesso de armazenamento



4.5 Solicitar listagem

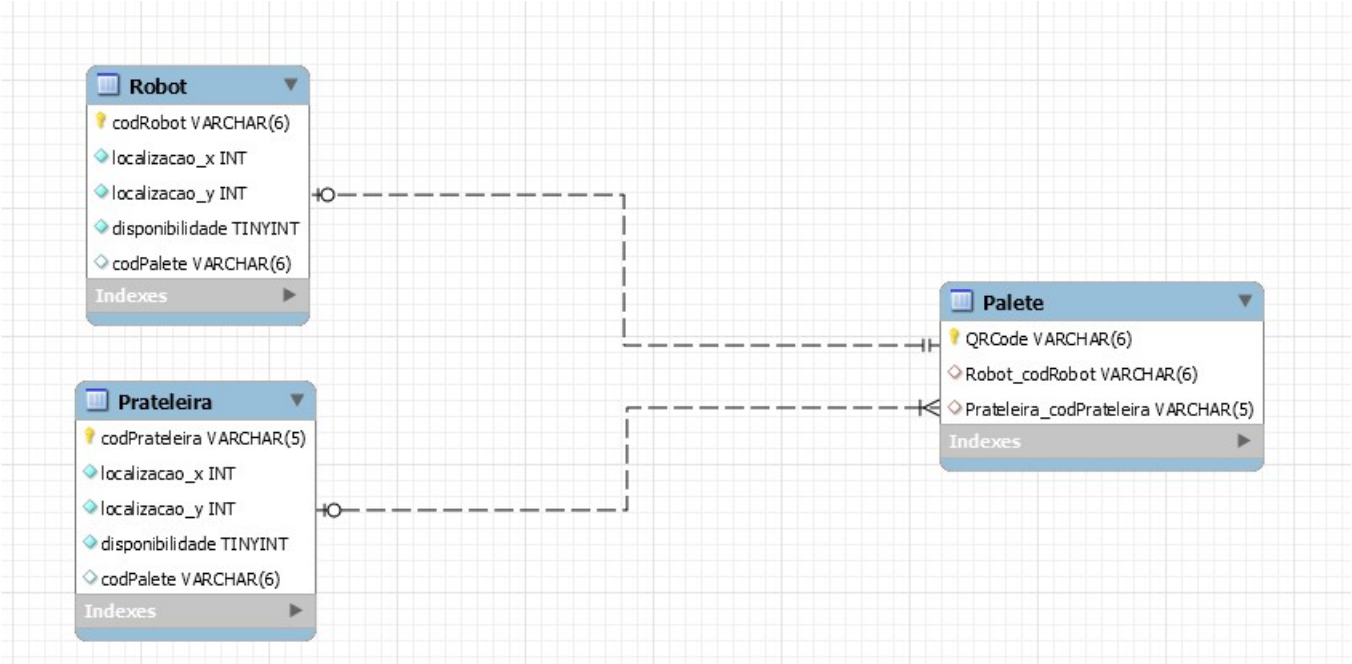


5 Diagrama de Packages



6 Modelo Lógico da Base de Dados

Com base na análise previamente feita, tendo em conta os Use Case e diagramas de sequência apresentados, bem como o diagrama de Classes, e após alguma discussão entre o grupo, chegámos à conclusão que a nossa base de dados deveria apresentar o seguinte modelo:



7 Descrição da Implementação

Nesta secção será apresentada a lógica por detrás da implementação de cada um dos componentes, sendo apresentado no final um diagrama com toda a implementação.

7.1 Implementação da Base de Dados

Tendo por base o modelo lógico acima apresentado, e achando somente necessário que na base de dados ficasse armazenada essa informação, procedemos à implementação dos DAOs relativos à Prateleira, ao Robot e à Palete.

Em cada um estes decidimos trabalhar com um Map em que a key é o código de cada um, nomeadamente codPalete, codRobot e QRCode, e o seu value é a classe respetiva que contém as informações armazenadas na base de dados e alguns atributos que são trabalhados em memória visto que essa informação é tratada somente em métodos implementados, não sendo informação pertinente que deva ser armazenada na base de dados em si.

Procedemos então à implementação dos vários métodos necessários para que pudessemos trabalhar com a base de dados, usando para tal o tratamento da informação com base em MySQL.

Sempre que pretendemos aceder à base de dados para obter, adicionar, remover ou alterar informação estabelecemos uma conexão com o servidor, de forma a facilitar-nos o acesso. Para podermos efetuar estas operações, como dito anteriormente, usamos a linguagem SQL.

7.2 Implementação da Lógica de Negócio

Tendo por base o diagrama de classes acima apresentado, bem como os Use Cases e diagramas de sequência, procedemos à implementação da camada da lógica de negócio, existindo nela seis classes: GPS, LeitorQrCode, Palete, Percurso, Prateleira, Robot, Sistema; e uma interface, SistemaFacade, que estabele a comunicação entre o modelo e o controlador.

7.2.1 Percurso

Esta classe, afeta ao Robot, destina-se a informar cada Robot sobre o percurso que deve realizar de forma a proceder ao transporte, recolha ou entrega de paletes.

É trabalhada em memória uma vez que considerámos desnecessário armazenar algo que depois de concluída a tarefa, desaparece.

Contém em si, uma lista de localizações que deve percorrer, bem como a distância total que percorrerá ao efetuar esse percurso.

7.2.2 GPS

Esta classe serve como identificador da localização das 3 principais classes neste trabalho, Palete, Robot e Prateleira, sendo também trabalhada em memória. No entanto as coordenadas necessárias para trabalhar com esta classe são armazenadas na base de dados.

Possui em si dois inteiros, correspondentes às coordenadas X e Y, e nela estão incluídos métodos, por exemplo, que calculam o Percurso que o Robot deve efetuar com base no mapa do Armazém, na localização inicial e na localização final. Este Percurso calculado será aquele que percorre uma menor distância para chegar ao seu destino.

7.2.3 LeitorQrCode

Esta classe está encarregue de gerar os códigos de QR afetos a cada Palete, existindo nela para tal um método que gera uma String alfanumérica de seis carateres aleatórios.

7.2.4 Palete

Esta é uma das principais classes e das mais trabalhadas neste trabalho.

Nela existe um código identificativo, uma localização GPS e a descrição do material que carrega.

Já na base de dados, possui um relacionamento quer com o Robot quer com a Palete, para podermos averiguar onde está localizada.

7.2.5 Prateleira

Esta classe, juntamente com a Palete e o Robot, é uma das principais e das mais trabalhadas, contendo nela o seu código identificativo, a sua disponibilidade para armazenar ou não paletes, a sua localização, e o código da Palete que possa estar em si armazenada.

Já na base de dados possui somente um relacionamento com a Palete, de modo a podermos saber que palete está armazenada em que Prateleira.

7.2.6 Robot

Esta classe, juntamente também, como já dito previamente, com a Palete e a Prateleira, é uma das principais e mais trabalhadas. Em si estão incluídos

o seu código identificativo, a sua disponibilidade para transportar ou não paletes, a sua localização e o código da palete que possa estar a transportar.

Estão também incluídos métodos que visam simular a deslocação do Robot para recolha e entrega de paletes.

Na base de dados possui somente um relacionamento com a Palete, de modo a podermos saber que palete se encontra a transportar.

7.2.7 Sistema e SistemaFacade

Por fim, existe uma última classe que agrega todas as acima enunciadas constituindo assim a lógica de negócios do nosso trabalho. Nesta classe estão incluídas as instâncias representativas dos DAOs do Robot, da Palete e da Prateleira, trabalhadas, como dito antes, através de Maps, contém uma lista de paletes que se encontram à espera de ser armazenadas na zona de receção e contém um mapa com pontos essenciais para a atribuição de percursos aos robots.

Nesta são implementados os métodos motores do nosso trabalho: comunicaCodigoQR, comunicaOrdemDeTransporte, notificaRecolhaDePalete, notificaEntregaDePalete e consultaListagem. Estes são os métodos referentes aos cinco Use Cases apresentados no enunciado que deveriam ser implementados.

No método **comunicaCodigoQR** é gerado um código QR aleatório por parte do LeitorQRCode após ser pedido ao utilizador o input do material que a palete carrega, sendo posteriormente adicionada à lista de paletes à espera na zona de receção.

No método **comunicaOrdemDeTransporte**, caso haja paletes à espera na zona de receção para serem transportadas, o sistema procura o Robot mais próximo e disponível para esta tarefa, reservando-o, ou seja passa a estar indisponível para outra tarefa, e procura também a prateleira mais próxima e disponível, reservando-a também. Caso não haja paletes à espera de serem transportadas é emitida uma mensagem de aviso por parte do Sistema. Em caso de sucesso de reserva do Robot e de Prateleira, isto é, existe pelo menos um Robot e uma Prateleira disponíveis, é enviada uma mensagem por parte do Sistema que informa qual o Robot escolhido.

No método **notificaRecolhaDePalete**, caso haja paletes às quais já foi atribuída a sinalização para transporte, caso haja Robots disponíveis, é enviada uma mensagem por parte do Robot a informar que a recolha foi bem sucedida. Caso não haja paletes à espera de recolha é enviada uma mensagem de aviso.

No método **notificaEntregaDePalete**, caso haja paletes às quais já

foi atribuída a sinalização para recolha, caso haja Robots disponíveis, é enviada uma mensagem por parte do Robot a informar que a entrega foi bem sucedida. Caso não haja paletes à espera de entrega é enviada uma mensagem de aviso.

No final de cada um destes métodos são atualizadas as informações na base de dados.

Por fim, o método **consultaListagem** apresenta uma listagem de todas as paletes existentes no Sistema, apresentando o seu código, o seu material e a sua localização atual. Caso não existam paletes no Sistema, é apresentada uma mensagem de aviso ao utilizador.

Por último, o SistemaFacade, interface criada para comunicação entre o controlador do programa e o camada da lógica de negócio, contém em si a definição destes cinco métodos.

7.3 Implementação da Interface

Para a realização deste trabalho, não sendo necessário uma interface gráfica, ou seja, a informação seria apresentada no terminal visível e em texto, criamos somente duas classes Menu e TextUI.

Na classe Menu definimos como serão apresentados os menus de opções ao utilizador, enquanto que na classe TextUI fazemos a comunição com o controlador e obtemos a informação a apresentar ao utilizador, nomeadamente as mensagens provenientes da camada da lógica de negócios.

```
 <<< Menu >>>
1 - Comunicar código QR
2 - Sistema: Comunicar Ordem de Transporte
3 - Notificar Recolha de Paletes
4 - Notificar Entrega de Paletes
5 - Gestor: Consultar listagem de localizações
0 - Sair
Opção:
```

```
 <<< Menu >>>
1 - Comunicar código QR
2 - Sistema: Comunicar Ordem de Transporte
3 - Notificar Recolha de Paletes
4 - Notificar Entrega de Paletes
5 - Gestor: Consultar listagem de localizações
0 - Sair
Opção: 1
Insira o nome do Produto: Batatas
Sistema:> Código Gerado com Sucesso! : nVKQFu
```

```
Opção: 2

 <<< Menu >>>
1 - kxvgTc: Arroz
2 - nVKQFu: Batatas
0 - Sair
Opção: 1
Sistema:> Robot Escolhido: R01
```

```
Opção: 3

 <<< Menu >>>
1 - kxvgTc: Arroz
0 - Sair
Opção: 1
Robot R01:> Palete Recolhida com Sucesso!
```

```
Opção: 4

 <<< Menu >>>
1 - R01: kxvgTc, Arroz
0 - Sair
Opção: 1
Robot R01:> Transporte Feito com Sucesso
```

```
 <<< Menu >>>
1 - Comunicar código QR
2 - Sistema: Comunicar Ordem de Transporte
3 - Notificar Recolha de Paletes
4 - Notificar Entrega de Paletes
5 - Gestor: Consultar listagem de localizações
0 - Sair
Opção: 5
Sistema:>
Palete { código: kxvgTc, material: Arroz, localização: 0-0, GPS: GPS{x=0, y=0} }
```

7.4 Implementação Final

Dado todos os pontos acima enunciados, passamos então a apresentar um modelo daquilo que é a nossa implementação final deste trabalho.

8 Análise de resultados

Com base no enunciado dado para a terceira fase, o grupo sentiu a necessidade de fazer algumas mudanças a nível dos diagramas de sequência e dos Use Cases, sendo que algumas poderão ter sido mais radicais que outras, acreditando no entanto que estas alterações foram necessárias para o bom funcionamento do nosso programa e de forma a cumprir com os requisitos presentes no mesmo enunciado.

Fazendo então uma reflexão crítica daquilo que o nosso projeto é capaz de fazer, e usando todos os pontos acima mencionados como justificação, acreditamos que o mesmo se encontra capaz de realizar o pretendido, implementando com sucesso a utilização de uma base de dados de forma completa e funcional, juntamente com a camada lógica de negócio e a camada da interface (vista).

Todos estes feitos culminam então num projeto capaz de responder aos requisitos no enunciado, e apesar de desde a primeira fase deste trabalho terem vindo a ser alterados certos aspetos da idealização do mesmo, os resultados apresentados são evidentes, o projeto é funcional e viável.

9 Conclusão

Este trabalho foi enriquecedor, na medida em que aprendemos a maneira correta de estruturar um projeto, seguindo determinados passos tais como: o estudo do domínio do problema e modelação do mesmo, o uso de Use Cases para obter uma noção mais clara sobre as interações dos diversos atores com o sistena, o uso de diagramas de sequência para aproximarmos os use cases da implementação final do projeto e o uso de diagramas de packages para uma melhor gestão das diversas classes.

Na primeira fase do projeto foi efetuada uma analise de requisitos onde averiguamos tudo sobre o problema proposto de maneira a consolidarmos ideias sobre o correto funcionamento do sistema, sendo-nos apresentados vários cenários a que o sistema deveria ser capaz de responder. Assim, após a avaliação destes cenários foi idealizado o Modelo de Domínio. De seguida tendo este modelo definido e consolidado, abordamos os requisitos funcionais do problema. Para isso, foi desenvolvido o diagrama de Use Case, onde foram identificados os diversos atores do sistema e a implicação que cada ação deste deveria exigir ou alterar do/no Sistema.

Na segunda fase do projeto começamos com o desenvolvimento dos diagramas de sequência para os Use Case criados na primeira fase, o que nos permitiu visualizar as interações entre o ator e o sistema relativamente a cada operação que ocorre no mesmo. Assim, partimos para a identificação das entidades passíveis de se tornarem classes na modelação do nosso sistema e, com elas, desenvolvemos o diagrama de classes. Após a definição de todas as classes e criação do diagrama de classes, conseguimos organizá-las melhor num diagrama de packages de modo a distribuir o trabalho necessário a ser realizado.

Na terceira fase do projeto, foi nos então pedido para implementarmos o nosso sistema, atendendo a uma diminuição de use cases (e por ventura diagramas de sequência) a considerar, procurando seguir a implmentação estipulada nas fases anteriores, porém ou detetamos algumas falhas e/ou melhores maneiras de desenvolver certa função do sistema, ou mesmo devido às restriçoes pedidas a esta fase, vimo-nos assim com necessidade de alterar alguns Use cases e diagramas de sequência previamente definidos. Com esta fase desenvolvemos também as capacidades de implementar DAOs, objetos de conexão a uma base de dados, tendo se mostrado talvez o nosso maior objetivo.

Podemos tomar como exemplo da implementação bem sucedida o Use Case de Solicistar Listagem, por parte do Gestor, onde ele solicita ao sistema a listagem de todas as paletes presentes no sistema e as suas localizações, e

apresenta-as na interface visual.

Posto isto, conseguimos criar um projeto funcional respeitador do "desenhado" nas fases anteriores, sendo capaz de lidar com mais do que um robot, implementando com sucesso as respetivas conexões à base de dados e tudo o que é referente a alterações e/ou atualizações na mesma, sendo capazes inclusivé de manter um estado previamente guardado das tabelas.