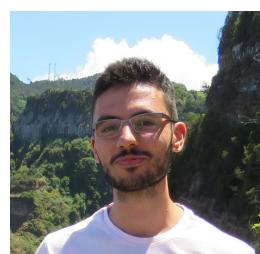




Universidade do Minho
Mestrado Integrado em Engenharia Informática
Unidade Curricular de Desenvolvimento de Sistemas de Software
Ano Letivo 2020/2021

***Sistema de gestão de stocks de um armazém
de uma fábrica***



Angélica Cunha
A84384

Ana César
A86038

José Gomes
A82418

Rui Chaves
A83693

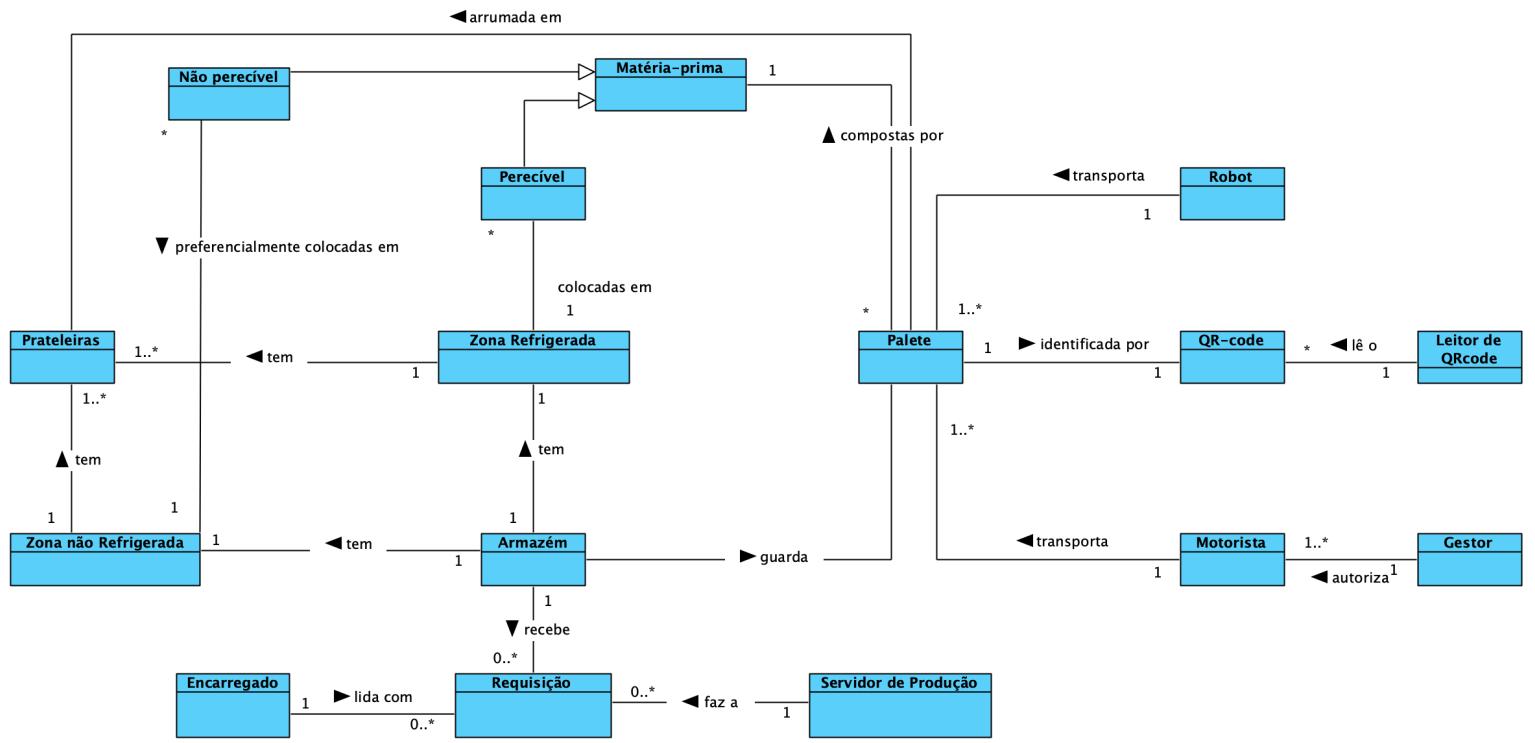
Grupo 38

Introdução

O presente relatório diz respeito à 2^a fase do desenvolvimento do trabalho prático da UC de DSS, em que o objetivo está centrado na modelação conceptual da nossa solução.

Começamos por fazer a arquitetura conceptual do sistema, de acordo não com o nosso conjunto de Use Case apresentado na 1^a fase deste trabalho, mas com o que nos foi fornecido pelos docentes após a entrega da 1^a fase. Foram para isso desenvolvidos os diagramas de sequência, de componentes e de classes.

Modelo de Domínio



Modelo de Use Case

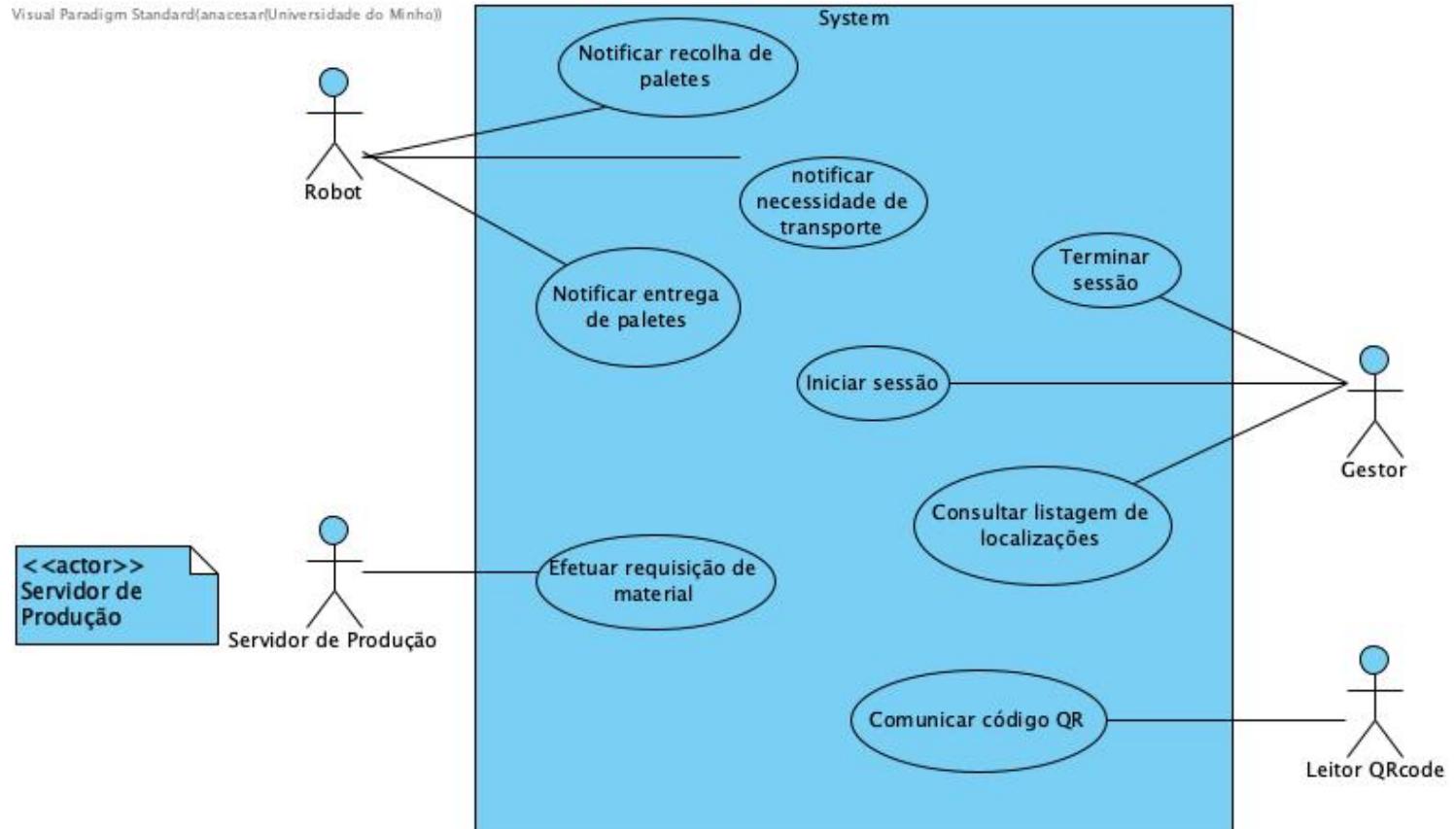


Figura 1 - Modelo de Use Case

O modelo de Use Case foi alterado consoante os objetivos pretendidos nesta segunda fase do projeto. Para os diferentes atores foram revistos os use cases a considerar, em relação ao gestor os use cases que passamos a considerar foram *Iniciar sessão*, *Terminar sessão* e *Consultar listagem de localizações*. Alteramos ainda os uses cases do Leitor QRcode que passa agora a estar associado a um novo use case, *Comunicar código QR*. Os restantes não sofreram alterações, contudo deixamos de considerar para esta fase os atores Encarregado e Motorista.

USE CASES

1. Iniciar Sessão

Descrição	Utilizador inicia sessão no sistema
Cenários	Gestor chega ao seu posto de trabalho e inicia sessão no sistema
Pré-condição	Utilizador ainda não ter sessão iniciada no sistema
Pós-condição	Gestor está autenticado
Fluxo Normal	<ol style="list-style-type: none"> 1) Sistema pede username ao gestor 2) Gestor introduz o seu username 3) Sistema pede password ao gestor 4) Gestor introduz password 5) Sistema valida username e password
Fluxo Alternativo 1	<p>[Password inválida] (Passo 5)</p> <p>5.1) Sistema informa que a palavra-passe introduzida não coincide com ID introduzido</p> <p>5.2) Regressa ao passo 4</p>
Fluxo Alternativo 2	<p>[Username inválido] (Passo 5)</p> <p>5.2.1) Sistema informa que ID não é valido</p> <p>5.2.2) Regressa ao passo 1</p>
Fluxo de Exceção 1	<p>[Password Inválida](Passo 5.1 – Fluxo alternativo 1)</p> <p>5.1.1) Gestor cancela início de sessão</p>

1º Dividir os fluxos

Fluxo Normal: <1, 2, 3, 4, 5>

Fluxos Alternativos: <5.1, 5.2> <5.2.1, 5.2.2>

Fluxo de Exceção: <5.1.1>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>GestãodeStocksLN</u> Validar username e password <1, 2, 3, 4, 5> Cancelar login FE(1) <5.1.1>	<u>GestãodeStocksLN</u> <code>+validaLogin(codUsername:String, password:String):Boolean</code> <code>+cancelaLogin()</code>

Neste use case é de realçar que os fluxos alternativos não dão origem a métodos presentes na camada lógica de negócio uma vez que se trata de interações do sistema com o utilizador, nomeadamente mensagens de erro possíveis na tentativa de iniciar sessão.

4º Agrupar os métodos em subsistemas:

Interface Utilizador	Interface Utilizador
+cancelaLogin()	+validaLogin(codUsername:String, password:String):Boolean

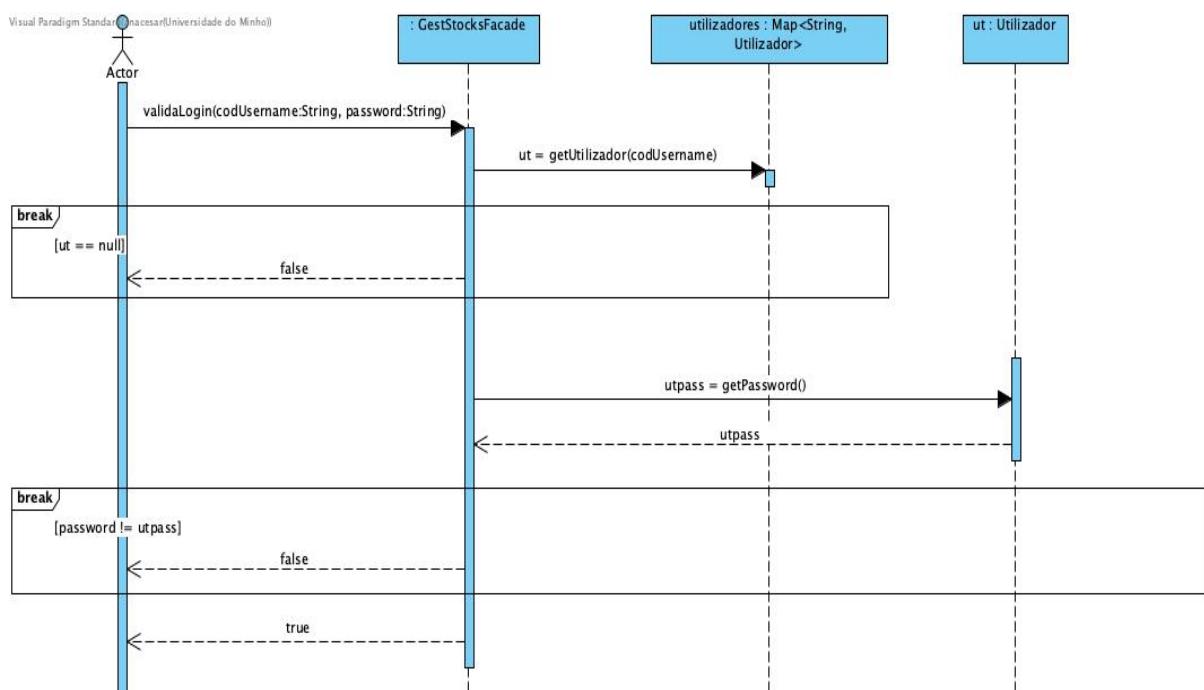


Figura 2 - Diagrama de Sequência "validaLogin"

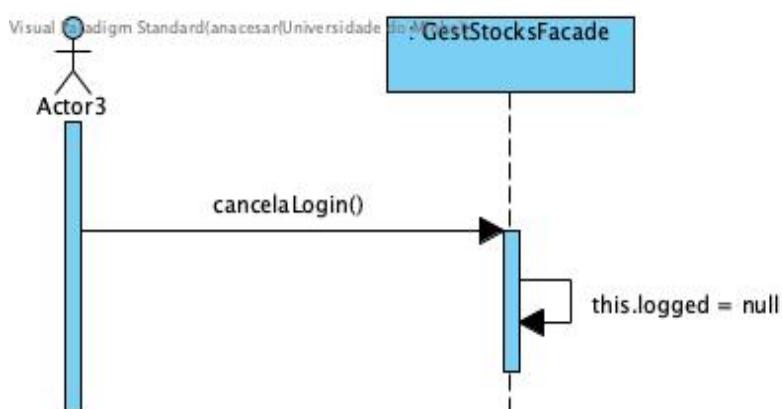


Figura 3- Diagrama de Sequência "cancelaLogin"

2. Terminar Sessão

Descrição	Gestor termina sessão no sistema
Cenários	Gestor termina sessão no sistema
Pré-condição	Gestor tem sessão iniciada no sistema
Pós-condição	Gestor não tem sessão iniciada no sistema
Fluxo Normal	<ul style="list-style-type: none"> 1) Gestor pretende terminar sessão 2) Sistema termina sessão 3) Sistema indica que sessão foi terminada

1º Dividir os fluxos

Fluxo Normal: <1, 2, 3>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de StocksLN</u> terminar sessão <1, 2, 3>	<u>Gestão de StocksLN</u> +terminarSessao()

4º Agrupar os métodos em subsistemas:

Interface Utilizador
+terminarSessao()

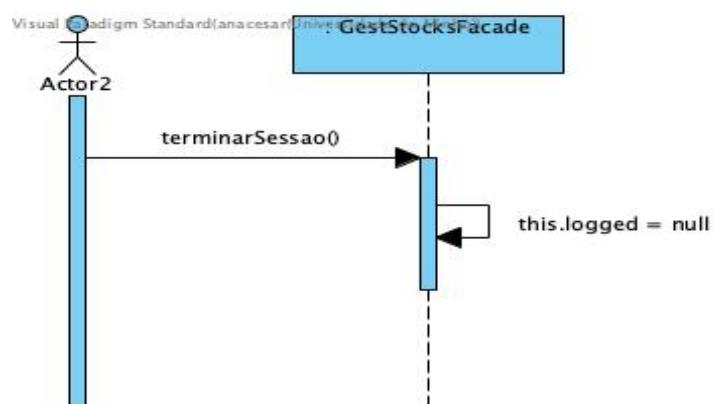


Figura 4 - Diagrama de Sequência de "terminarSessao"

3. Consultar listagem de localizações

Descrição	Gestor consulta a lista de localizações de determinadas paletes no armazém
Cenários	Há necessidade de consultar a localização de todas as paletes necessárias a uma requisição; Gestor pretende saber localização de determinadas paletes
Pré-condição	É feita uma requisição de material ao sistema
Pós-condição	Gestor tem o conhecimento das localizações pretendidas
Fluxo Normal	<p>1) Gestor informa o sistema quais os códigos das paletes a localizar</p> <p>2) Sistema verifica para cada palete a validade do seu código e associa-lhe a sua devida localização</p> <p>3) Sistema devolve ao Gestor as localizações associadas às paletes</p>
Fluxo Alternativo 1	<p>[Código de paleta inválido] (Passo 2)</p> <p>2.1) Sistema associa ao código da paleta uma mensagem de código inválido</p> <p>2.2) Regressa a 2</p>

1º Dividir os fluxos

Fluxos Normais: <1, 2, 3>

Fluxos Alternativos: <2.1, 2.2>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de StocksLN</u> verifica códigos e devolve as localizações das paletes	<u>Gestão de StocksLN</u> <code>+localizacoes(paletes : List<String>) : Map<String, Localizacao></code>

4º Agrupar os métodos em subsistemas:

Interface Inventory <code>+localizacoes(paletes : List<String>) : Map<String, Localizacao></code>

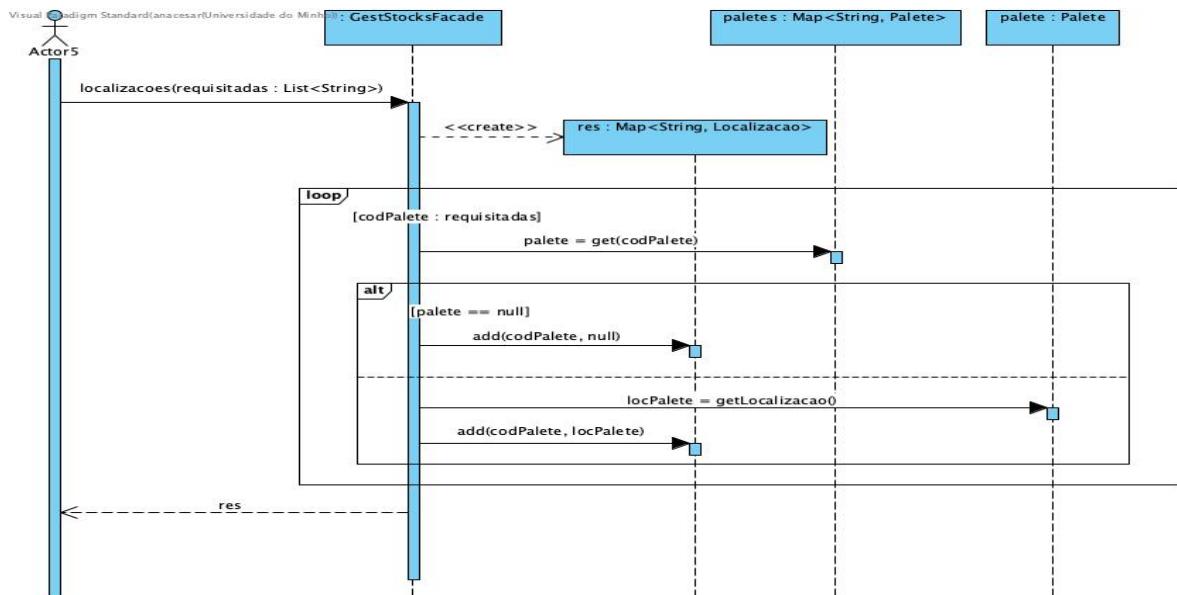


Figura 5 - Diagrama de Sequência "localizacoes"

4. Notificar necessidade de transporte

Descrição	O sistema notifica um robot disponível a necessidade de transporte de paletes dentro do armazém
Cenários	Sistema seleciona robot e calcula a rota que tem a efetuar
Pré-condição	Existem paletes no armazém para descolar
Pós-condição	O robot fica notificado, sabendo qual a rota a efetuar e paletes a deslocar
Fluxo Normal	<ol style="list-style-type: none"> 1) Sistema procura robot para transportar palete 2) Sistema encontra robot para transporte 3) Sistema calcula rotas para o robot 4) Sistema <i>comunica rotas e paleta ao robot</i> 5) Sistema regista paleta na lista de paletes a aguardar transporte
Fluxo de Exceção 1	[Não existe nenhum robot disponível] (Passo 1) 1.1) Sistema marca pedido de transporte como irrealizado

1º Dividir os fluxos

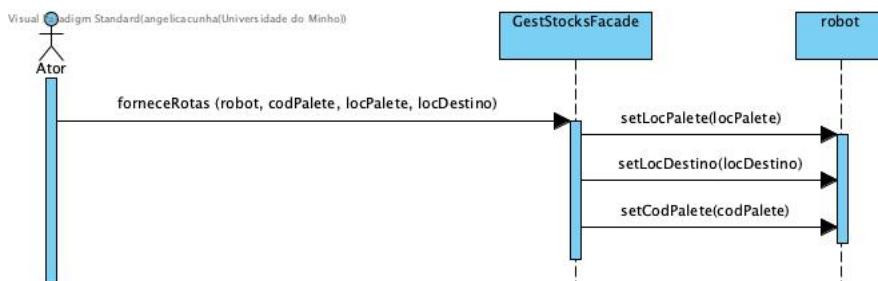
Fluxos Normais: <1,2> <3,4> <5>

Fluxos Alternativos: <2.1, 2.2>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de Stocks LN</u> Procura robot disponível <1,2> Dá rota ao robot <3,4> Adicionar paleta à lista de paletes a aguardar transporte <5>	<u>Gestão de Stocks LN</u> +getRobot (locPaleta: Localização): Robot +forneceRotas (robot: Robot, codPaleta: String, locPaleta: Localização, locDestino: Localização) +transportarPaleta(codPaleta: String)

4º Agrupar os métodos em subsistemas:

Interface Inventário	Interface Robot
+getRobot (locPaleta: Localização): Robot +transportarPaleta(codPaleta: String)	+forneceRotas (robot: Robot, codPaleta: String, locPaleta: Localização, locDestino: Localização)



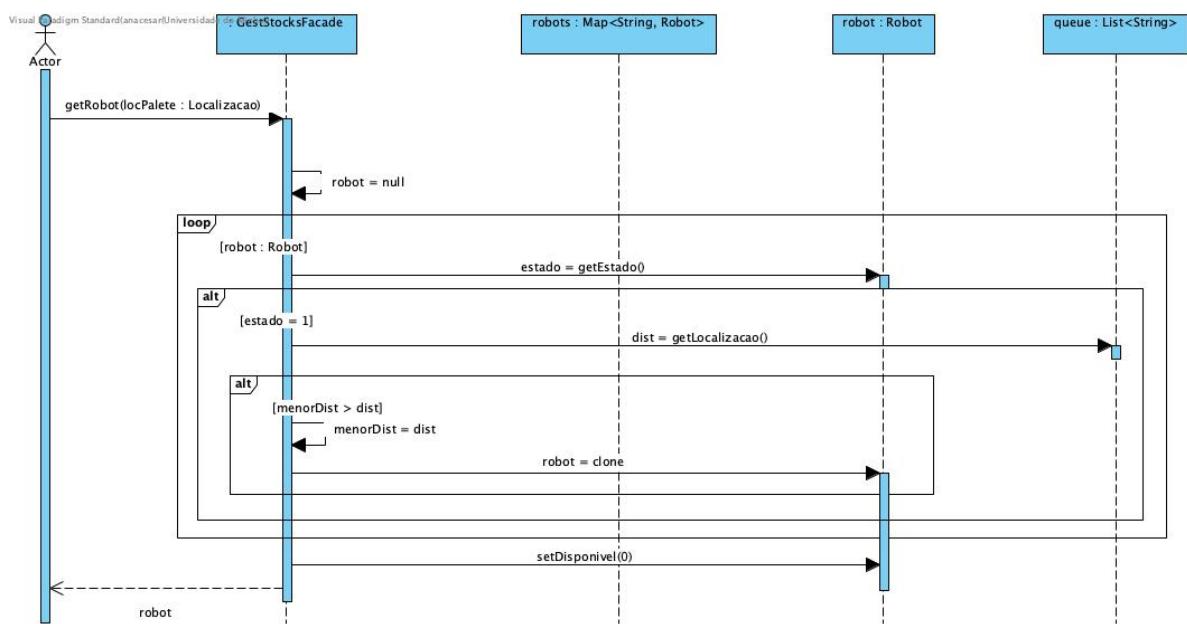


Figura 7- Diagrama de Sequência "getRobot"

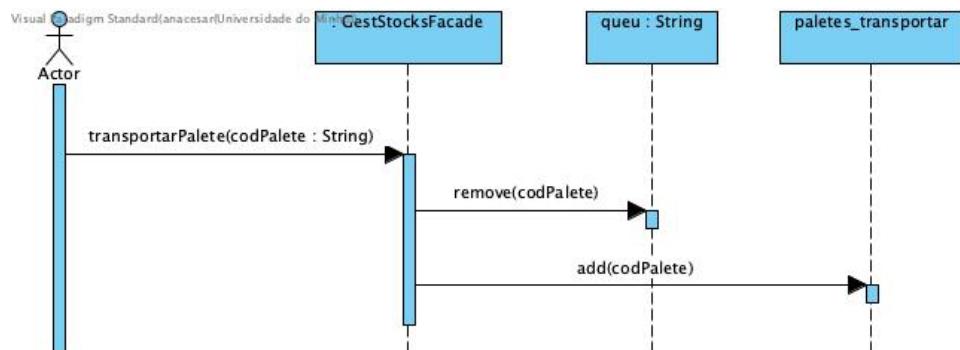


Figura 8- Diagrama de Sequência de "transportarPalete"

5. Notificar recolha de palete

Descrição	Robot notifica que recolheu palete
Cenários	Robot vai buscar uma palete para armazenar
Pré-condição	Robot sabe qual a palete a recolher e rota a efetuar
Pós-condição	Paleta recolhida
Fluxo Normal	<ul style="list-style-type: none"> 1) Robot desloca-se à localização da paleta 2) Robot recolhe paleta 3) Robot notifica o Sistema da recolha da paleta
Fluxo de Exceção 1	<p>[Paleta não se encontra na localização dada] (Passo 3)</p> <ul style="list-style-type: none"> 3.1) Robot notifica o sistema de que a paleta não se encontra na localização devida 3.2) Sistema coloca paleta na lista de paletes extraviadas

1º Dividir os fluxos

Fluxos Normais: <1, 2, 3>

Fluxos de Exceção: <3.1, 3.2>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de Stocks LN</u> Robot notifica sistema que recolheu paleta Sistema coloca paleta na lista de paletes extraviadas	<u>Gestão de Stocks LN</u> +paletaRecolhida(robot : Robot, codPaleta: String) +paletaExtraviada(codPaleta: String)

4º Agrupar os métodos em subsistemas:

Interface Robot	Interface Inventario
+paletaRecolhida(codPaleta: String)	+paletaExtraviada(codPaleta: String)

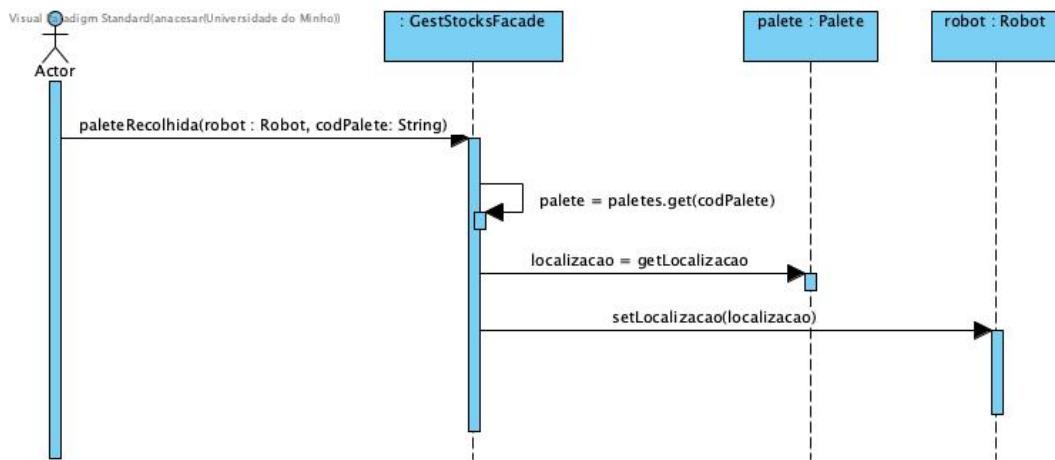


Figura 9 - Diagrama de Sequência de "paletaRecolhida2"

6. Notificar entrega de palete

Descrição	Robot notifica que entregou palete
Cenários	Robot entrega uma palete à zona de entregas
Pré-condição	Ter uma palete para entrega
Pós-condição	Robot fica livre
Fluxo Normal	<ol style="list-style-type: none"> 1) Robot desloca-se à localização de entrega da paleta 2) Robot coloca a paleta no destino 3) Robot notifica o Sistema que entregou a paleta 4) Sistema atualiza localização da paleta

1º Dividir os fluxos

Fluxos Normais: <1, 2, 3> <4>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de Stocks LN</u> Notificação de entrega de paleta <1, 2, 3> Atualização da localização da paleta <4>	<u>Gestão de Stocks LN</u> <code>+paleteEntregue(robot : Robot, codPaleta: String)</code> <code>+updateLocalizacao(codPaleta : String, locDestino : Localizacao)</code>

4º Agrupar os métodos em subsistemas:

Interface Robot	Interface Inventario
<code>+paleteEntregue(codPaleta: String)</code>	<code>+ updateLocalizacao(codPaleta : String, locDestino: Localizacao)</code>

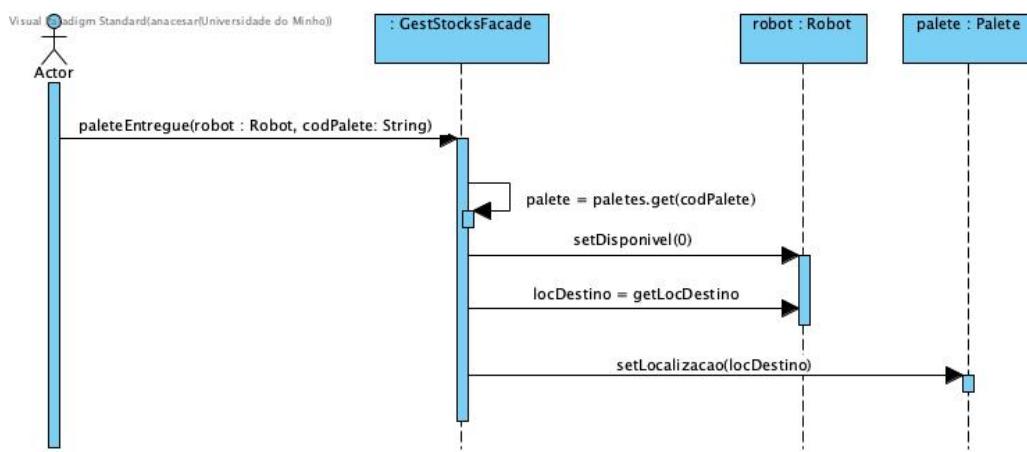


Figura 10 - Diagrama de Sequência de "paleteEntregue"

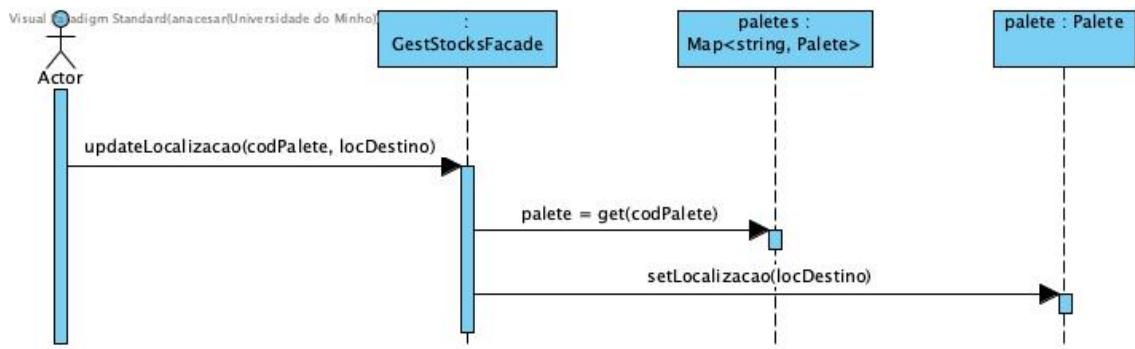


Figura 11 - Diagrama de Sequência de "updateLocalizacao"

7. Efetuar requisição de material

Descrição	O servidor de produção faz requisição de material.
Cenários	Servidor de produção requisita 3 paletes de material
Pré-condição	TRUE
Pós-condição	O sistema fica com o registo das paletes que foram requisitadas.
Fluxo Normal	<p>1) O servidor de produção comunica quais as paletes necessárias</p> <p>2) Sistema valida a disponibilidade das paletes</p> <p>3) Sistema cria registo das paletes a entregar</p>
Fluxo Alternativo 1	[Alguma palete não disponível] (Passo 2) <p>2.1) Sistema comunica ao servidor de produção as paletes que não estão disponíveis de momento</p> <p>2.2) Servidor de produção pede cancelamento das paletes em falta</p> <p>2.3) Sistema cancela paletes em falta.</p> <p>2.4) Regressa a 3)</p>
Fluxo Alternativo 2	[Pedido por fases] (Passo 2.2 – Alt.1) <p>2.2.1) Servidor de produção confirma pedido total</p> <p>2.2.2) Sistema cria registo de paletes em falta para entrega posterior</p> <p>2.2.3) Regresso a 3)</p>
Fluxo de Exceção 1	[O pedido não pode ser parcial] (passo 2.2 – Alt.1) <p>2.2.1) Servidor de produção cancela o pedido</p>

1º Dividir os fluxos

Fluxo Normal: <1 ,2> <3>

Fluxos Alternativos: <2.1, 2.2, 2.3> <2.2.1, 2.2.2>

Fluxo de Exceção: <2.1.1>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
<u>Gestão de StocksLN</u> valida disponibilidade de paletes <1, 2> cria registo paletes a entregar <3> cancela paletes em falta FA(1) <2.1, 2.2, 2.3> pedido por fases FA(2) <2.2.1, 2.2.2> cancela o pedido FE(1) <2.1.1>	<u>Gestão de StocksLN</u> +validaPaletes(paletes : List<String>): Requisicao +registaRequisicao(req: Requisicao) +pedidoFases(req : Requisicao) : Requisicao +cancelaPedido(req : Requisicao)

4º Agrupar os métodos em subsistemas:

Interface Inventario	Interface Utilizador
<pre>+validaPaletes(paletes : List<String>): Requisicao +registaRequisicao(req : Requisicao)</pre>	<pre>+pedidoFases(req : Requisicao) : Requisicao +cancelaPedido(req : Requisicao)</pre>

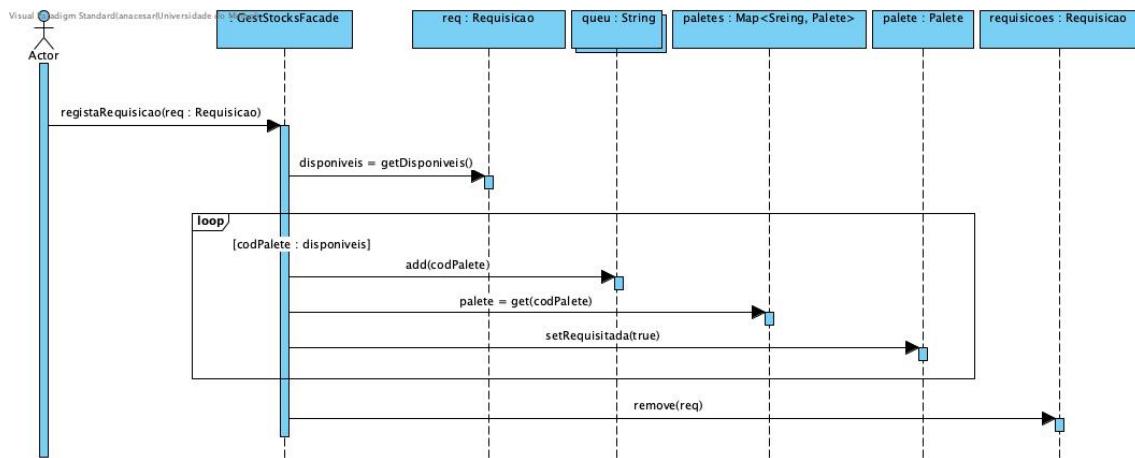


Figura 12 - Diagrama de Sequência de "registaRequisicao"

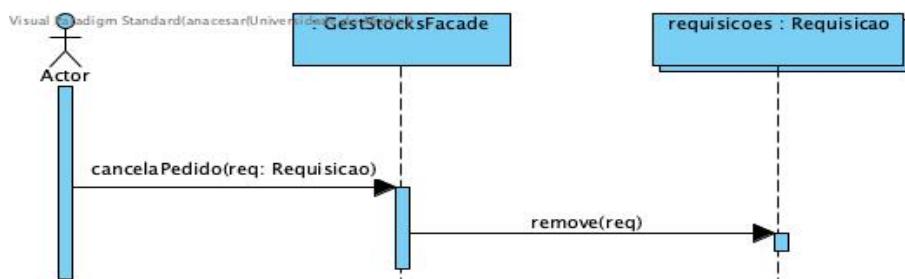


Figura 13 - Diagrama de Sequência de "cancelaPedido"

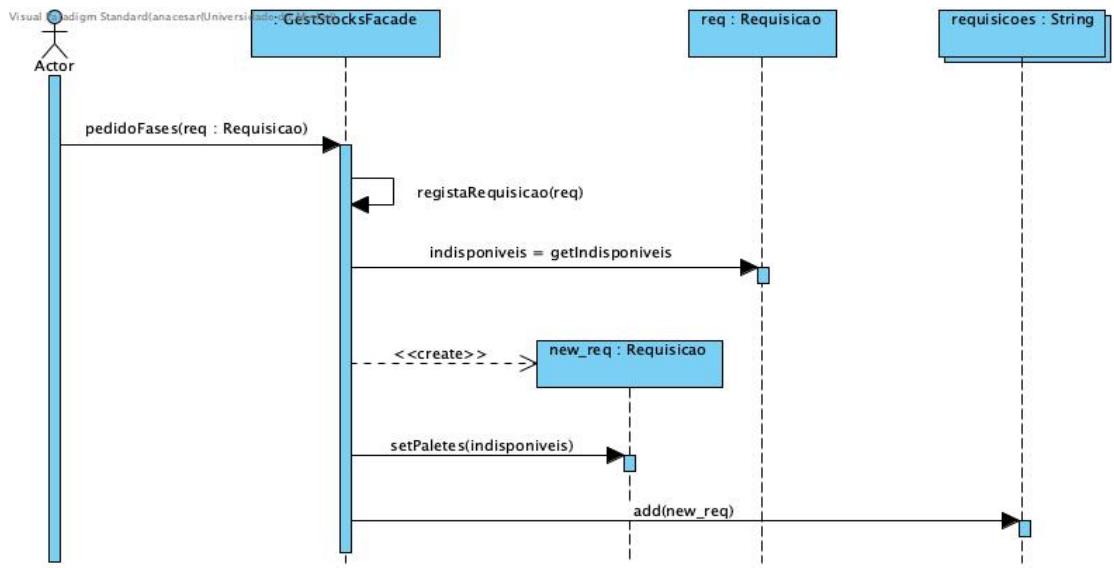


Figura 14 - Diagrama de Sequência de "pedidoFases"

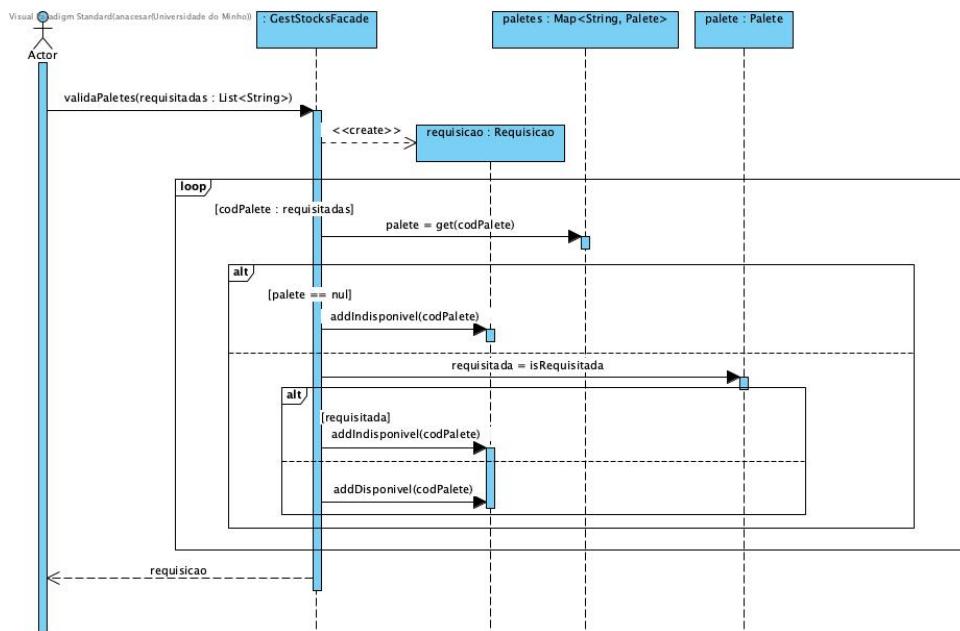


Figura 15 - Diagrama de Sequência "validaPaletes"

8. Comunicar código QR

Descrição	Leitura de código QR-code da palete e comunicação ao sistema
Cenários	Chegada de paletes por registar à receção do armazém.
Pré-condição	Existem paletes na zona de descarga
Pós-condição	Paleta registada e presentes na zona de receção à espera de armazenamento
Fluxo Normal	<ol style="list-style-type: none"> 1) Leitor QR-code lê código QR-code 2) Leitor QR-code comunica código associado ao sistema 3) Sistema regista paleta associada ao seu código 4) Sistema regista código de paleta na fila de paletes a aguardar armazenamento
Fluxo Alternativo 1	[Leitor QR-code não consegue ler código] (Passo 1) <ol style="list-style-type: none"> 1.1) Gestor introduz manualmente código da paleta 1.2) Regressa a 2

1º Dividir os fluxos

Fluxo Normal: <1, 2, 3 ,4>

Fluxos Alternativos: <1.1, 1.2>

2º Identificar responsabilidades da lógica de negócio	3º Identificar os métodos
Registo da palete <1, 2, 3 ,4>	+registarPalete(codPalete : String)

4º Agrupar os métodos em subsistemas:

Interface Inventario
+ registrarPalete(codPalete : String)

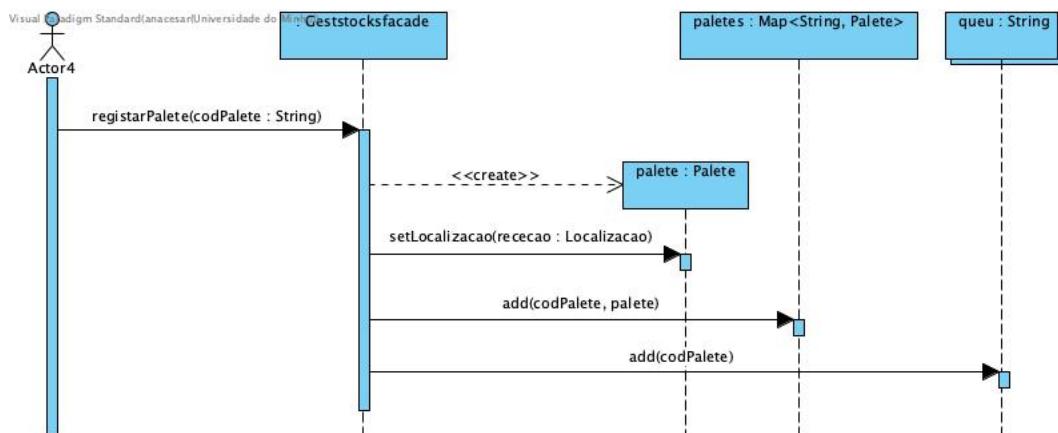


Figura 16 - Diagrama de Sequência "registarPalete"

Diagrama de Classes

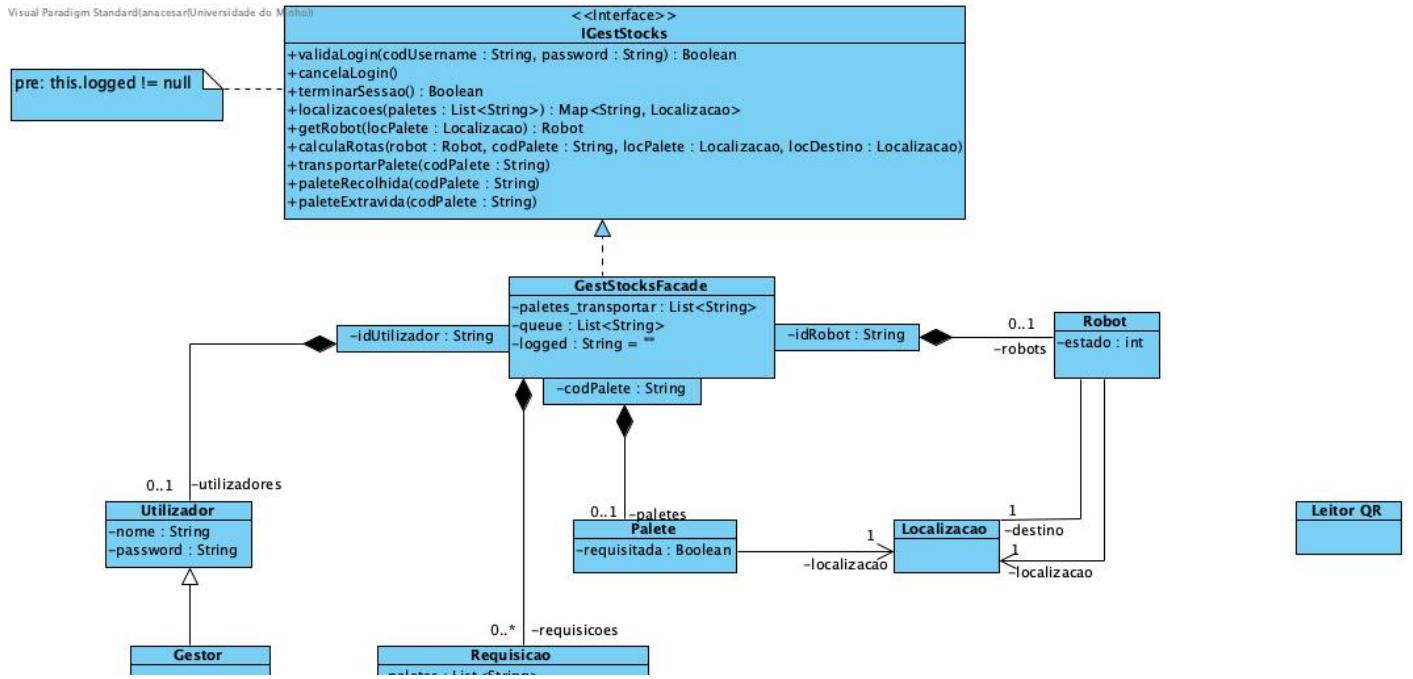


Figura 17 - Diagrama de Classes

Diagrama de Componentes

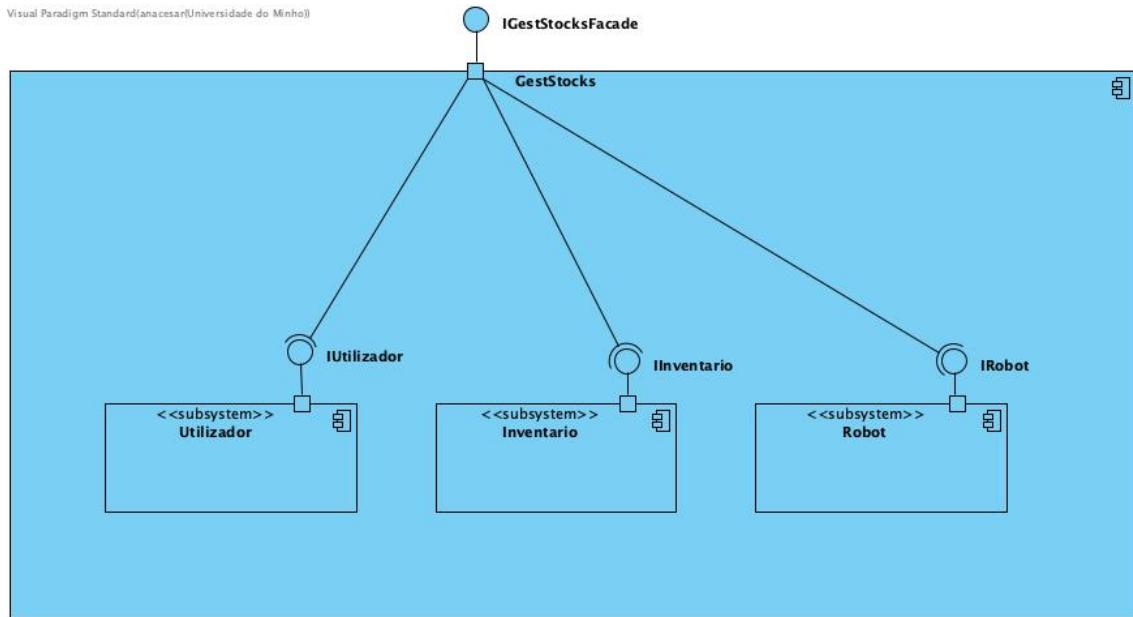


Figura 18 - Diagrama de Componentes

Análise Crítica

A resolução da 2^a fase deste projeto foi demorosa, mas iterativa. Identificamos os Use Cases que não incluímos na entrega da 1^a fase e reavaliarmos a Análise de Requisitos. Fizemos também as devidas alterações nos modelos de Use Case e de Domínio. Para cada Use Case enumeramos os fluxos, identificamos as responsabilidades da Lógica de Negócio e identificamos os métodos a estas associadas. Com a identificação da API global da Lógica de Negócio, foi possível dividir e categorizar em três APIs parciais e subsistemas que implementam as interfaces. Identificar os subsistemas foi relativamente simples, pelo que não fizemos alterações ao diagrama de Componentes. Ao passar para a realização dos modelos de classe e de sequência, tarefa feita em paralelo, os métodos foram alvos de algumas alterações. Estas constantes revisões nos métodos e nos diagramas fizeram com que esta parte do trabalho tenha sido a mais demorada.