



Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia Informática

Unidade Curricular de Desenvolvimento de Sistemas de Software

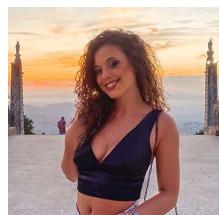
Ano Letivo de 2021/2022

Sistema de apoio a departa- mento técnico

Grupo 31



62608 - Marco Sousa



93198 - Mariana Marques



93271 - José Malheiro



94269 - Miguel Fernandes

3 de Janeiro de 2022

DSS

Resumo

A utilização de modelos auxilia a compreensão do problema, simplifica a comunicação de ideias e documenta as decisões tomadas durante o desenvolvimento.

Um centro de reparações encontra problemas diários que limitam a sua capacidade de resposta, diminuindo a eficiência dos colaboradores e impactando no volume de trabalho concretizado.

Através da análise dos cenários de utilização, foi possível efetuar a modelação completa de um sistema que responderia a estes equívocos. Foi necessária a conceção de vários modelos e diagramas, tais como:

Modelo de Domínio Para representar as entidades e as suas relações

Diagrama de Use Cases Graficamente apresentar as funcionalidades criadas para o sistema

Diagr. de Componentes Com o intuito de estruturar a arquitetura do sistema

Diagrama de Classes Definindo concretamente as classes e relacionamentos estruturais a implementar

Diagramas de Sequência Caracterizar o comportamento dos métodos definidos para a implementação

A estratégia utilizada permitiu seguir uma linha de raciocínio estruturada e ao utilizar uma linguagem de modelação unificada obteve-se um conjunto de diagramas que são facilmente interpretáveis entre pares. Para além disso, permitiu agilizar o processo de implementação do sistema, provando-se uma estratégia extremamente útil.

Área de Aplicação: Análise de Requisitos

Palavras-Chave: *Unified Modeling Language (UML), Visual Paradigm, Modelação de Domínio, Modelação de Requisitos Funcionais, Diagrama de Use Case, Modelo de Domínio, Análise de Projeto, Conceção de Projeto, Análise de Requisitos*

Índice

1	Introdução	1
1.1	Contextualização	1
1.2	Justificação e Utilidade do Sistema	1
1.3	Breve Descrição do Enunciado Proposto	2
1.4	Objectivos	2
1.5	Estrutura do Relatório	2
2	Modelação de Domínio	4
2.1	Descrição do problema	4
2.1.1	Principais Alterações na Definição do Problema	5
2.2	Entidades	6
2.2.1	Cliente	6
2.2.2	Colaborador	7
2.2.3	Equipamento	7
2.2.4	Orçamento	8
2.2.5	Reparação	8
2.2.6	Forma de Contacto	9
2.2.7	Comunicação	9
2.2.8	Plano de Trabalhos	9
2.2.9	Passo de Reparação	9
2.2.10	Material	10
2.2.11	Principais alterações nas Entidades	10
2.3	Diagrama de Modelo de Domínio	10
2.3.1	Modelo de Domínio Anterior	11
2.3.2	Modelo de Domínio Atual	11
3	Modelação dos Requisitos Funcionais	13
3.1	Modelo de Use Cases	14
3.1.1	Principais Diferenças com a Primeira Fase	14
3.2	Especificação Use Cases	15
4	Modelação Conceptual	60
4.1	Diagrama de Componentes	60

4.2	Diagrama de Classes	61
4.2.1	GestReparaçõesFacade	69
4.3	Diagramas de Sequência	74
4.3.1	Fazer Orçamento	74
4.3.2	Realizar Reparação	76
4.3.3	Pedir Reparação Expresso	77
4.4	Diagrama de Package	78
5	Implementação	82
5.1	Menus	84
6	Considerações Finais	88
6.1	Conclusões	88
Anexos		90
	Anexo 1 - Cenários	90
	Diagramas de Sequência	92
	Diagrama de classe	140

Lista de Figuras

2.1	Modelo de Domínio	11
2.2	Modelo de Domínio	12
3.1	Diagrama de <i>Use Case</i>	15
4.1	Diagrama de Componentes	62
4.2	Diagrama de Classe - ReparacoesLN	64
4.3	SubSistema SSClientes 4.2	66
4.4	SubSistema SSReparacoes 4.2	68
4.5	SubSistema SSReparacoes 4.2	70
4.6	Precos	70
4.7	Custo Total Reparacao	71
4.8	Reparacoes Por Mes	71
4.9	Agenda	73
4.10	Diagrama de sequência getOrcamentosAtivos	74
4.11	Diagrama de sequência criarPasso(1)	75
4.12	Diagrama de sequência criarPasso(2)	75
4.13	Diagrama de sequência alterarEstadoOrc	75
4.14	Diagrama de sequência comunicarErro	76
4.15	Diagrama de sequência AlterarEstadoRep	76
4.16	Diagrama de sequência registrarPasso	77
4.17	Diagrama de sequência addRepExpresso(1)	78
4.18	Diagrama de sequência existeDisponibilidade	79
4.19	Diagrama de sequência addRepExpresso(2)	79
4.20	Diagrama de sequência temDisponibilidade	80
4.21	Diagrama de Package	81
5.1	Recorte do diagrama de Gantt	83
5.2	Menu inicial	84
5.3	Menu Técnico	84
5.4	Menu Fazer orcamento	85
5.5	Exemplo Adicionar Passo	85
5.6	Menu Funcionário Balcão	86

5.7	Exemplo registrar cliente	86
5.8	Exemplo registrar equipamento	86
5.9	Exemplo pedir orçamento	86
5.10	Exemplo confirmar orçamento	87
5.11	Exemplo pedir reparação expresso	87
5.12	Menu Gestor	87

1 Introdução

O presente relatório foi desenvolvido no âmbito da Unidade Curricular (UC) de Desenvolvimento de Sistemas de Software (DSS), tendo como principal objetivo apresentar uma possível conceção de um Sistema de Gestão para Centros de Reparação de Equipamentos Eletrónicos, ao qual iremos designar por *Techsupport*, doravante designado *aplicação*.

1.1 Contextualização

A procura por um serviço de reparação, compreende um atendimento célere e garantia de acompanhamento ao longo de todo o processo. Para que tal seja possível, foi proposto pela equipa docente o desenvolvimento de uma aplicação que permita essa gestão.

1.2 Justificação e Utilidade do Sistema

Todo o processo de atendimento e reparação de um equipamento, requere cuidado e organização, de maneira a garantir um bom serviço.

Em pequenos negócios locais, a aquisição deste sistema não seria tão vantajosa pois, como a procura seria pouca, seria fácil de gerir as interações com clientes, da mesma maneira, gerir os seus equipamentos e reparações. Porém, é sempre uma ajuda ter um sistema que registe todo o historial do negócio.

Por sua vez, quando se trata de um grande volume de clientes e bastantes equipamentos com variadas reparações, seria bastante útil adquirir este sistema devido às suas funcionalidades, como por exemplo, seria possível gerir toda a agenda dos técnicos, de forma a calendarizar todas as reparações normais ou do momento (expresso), verificando se é possível ou não, seria possível adicionar novas reparações e passos da mesma, tal como os seus preços, de maneira a evitar repetições e economizar tempo, aceder às listagens dos colaboradores subordinados, avaliando assim o seu trabalho, entre outras.

Em suma, verifica-se uma maior necessidade de aquisição do sistema *Techsupport* em grandes negócios, o que não impede sua aquisição em qualquer tipo de negócio, devido à sua utilidade.

1.3 Breve Descrição do Enunciado Proposto

Tal como referido, o enunciado propõe a conceção e implementação de uma aplicação e apresenta os cenários de utilização que deverão ser suportados por esta.

O desenvolvimento desta aplicação foi marcado por 2 fases, nomeadamente:

- Análise
- Modelação Conceptual e Implementação

Primeiramente, encontra-se a **Análise**, que inclui (i) a análise do domínio do problema, através da modelação de um modelo de domínio e (ii) de requisitos funcionais, onde se utilizará a modelação de requisitos funcionais, com uma visão orientada aos *use cases*.

Posteriormente, encontra-se a **Modelação Conceptual e Implementação**, que inclui (i) o diagrama de componentes e (ii) todo o processo de desenvolvimento do diagrama de classes, onde foram definidas todas as classes e métodos do nosso sistema, (ii) e o desenvolvimento dos diagramas de sequência que ilustram os métodos criados.

1.4 Objectivos

De forma breve, o grupo pretende desenvolver competências técnicas e conceptuais no desenvolvimento de modelos e diagramas, por forma a se preparar para o mercado de trabalho de Engenharia de Software.

1.5 Estrutura do Relatório

Pretende-se efetuar uma abordagem ao problema utilizando uma estratégia estruturada e sistemática, concretizando a modelação conceptual do problema e, agora, a sua implementação.

Assim, é obtido um **modelo de domínio** que fornece uma *framework* conceptual para ra-

ciocinar sobre o problema. Aqui, são capturadas as **Entidades** do problema e os **Relacionamentos** entre elas. Posteriormente, procurou-se analisar os requisitos, em particular os funcionais, pretendendo descrever o que o sistema deve fazer.

Na segunda fase deste projeto, primeiramente, foi realizado o **modelo de componentes**, concebendo um modelo direcionado a interfaces que utiliza a estratégia *Facade*. Aqui, iniciou-se todo o processo de desenvolvimento no **diagrama de classes**, que originou mudanças exaustivas no projeto realizado na primeira fase, e onde foram definidas as classes e métodos que compõem o sistema. Posteriormente, foram realizados os **diagramas de sequência** que ilustram os métodos desenvolvidos.

No final desta segunda fase, ocorreu a implementação de todo o sistema, que é demonstrada por testes que foram realizados.

Por fim, foi efetuada uma breve análise crítica do trabalho desenvolvido.

2 Modelação de Domínio

Numa tentativa de capturar os elementos intervenientes do problema, as entidades e o relacionamento entre eles foi necessária a criação de uma visão estática do mesmo, i.e. um modelo de domínio.

Com o auxílio deste será formulada uma *framework* conceptual do sistema, permitindo raciocinar sobre o problema e estabelecer o vocabulário a ser usado no decorrer do projeto.

2.1 Descrição do problema

O centro de reparações oferece dois tipos de intervenções:

Programada Uma intervenção de duração variável, em que o equipamento é reparado após a aprovação do cliente face a um orçamento criado.

Expresso Conjunto de serviços limitado, com preço fixo e que apenas é aceite mediante disponibilidade de tempo para realização imediata.

Para desenvolver o orçamento é necessário realizar uma avaliação do equipamento e definir um plano de trabalhos. Este consiste na sequência de passos da reparação, cada um apontando o custo, o material necessária e o tempo previsto da reparação.

Os colaboradores são os principais atores do sistema. O funcionário de balcão identifica o(s) cliente(s) e o(s) respetivo(s) equipamento(s). Posteriormente, efetua o pedido de orçamento. Assume-se que um cliente pode ter um ou mais equipamentos. Sendo que o equipamento pode avariar mais do que uma vez, torna-se relevante a sua entidade ser reutilizada entre cada reparação. Por isso, um equipamento tem um histórico de reparações.

O técnico é o responsável pela especificação do orçamento e realização da reparação. Por uma questão de otimização de tempo e recursos, um técnico pode estar associado a mais do que uma reparação, contudo, ao contrário do que foi definido na primeira fase do projeto, uma reparação não pode ser efetuada por mais do que um técnico. Neste sentido, uma reparação é feita, desde o início até ao fim por apenas um técnico.

Para o desenvolvimento de um orçamento, o técnico constrói um novo plano de trabalhos. Este, por sua vez, é constituído por um conjunto de passos de reparação.

Um passo de reparação possui o material que irá usar, sendo este único ao passo e relativo a todos os materiais que irá necessitar para o seu desenvolvimento. São permitidos passos que não usem materiais, *i.e.* uma formatação. De modo análogo à entrega da primeira fase, permanece-se associado a um passo de reparação o tempo necessário para o desenvolver e todos os sub-passos que engloba.

Está estipulado que apenas as reparações programadas possuem um plano de trabalhos a seguir e cada plano de trabalho é único à reparação a que foi associado. Neste sentido, as reparações expresso (que tem características particulares - serviço fixo), por serem imediatas, distinguem-se por não possuirem um conjunto de passos a definir, sendo a sua realização conhecida pelo técnico. Ambos os tipos de reparação mantêm-se associados à entidade **Reparação**.

O gestor terá um papel de administração, avaliando os desempenhos dos outros elementos e recrutando novos colaboradores (tendo de os adicionar ao sistema).

Por último, em casos em que seja requerido, um colaborador poderá comunicar com os clientes, a partir de um *SMS* ou *Email*.

2.1.1 Principais Alterações na Definição do Problema

No decorrer da realização do projeto, a partir da experiência de outros trabalhos semelhantes, ocorreram alterações ao nível da dinâmica do sistema definido na primeira fase, para traduzir numa aplicação mais concisa.

As principais e mais notórias alterações:

Realização de uma Reparação com vários Técnicos

Devido à complexidade na sua implementação, não será apresentada a possibilidade de realizar a mesma reparação por técnicos diferentes, sendo uma funcionalidade útil cuja implementação seria importante em iterações seguintes do sistema.

Categoría Material

Numa restruturação posterior, a categoria de material foi retirada do sistema, devido à sua implementação ambiciosa e trabalhosa. Apesar da existência de uma categoria de material permitir flexibilidade ao nível da definição de passos de reparação genéricos, foi decidido, após uma avaliação da situação do trabalho que não seria implementada esta dinâmica no sistema nesta iteração. Contudo, permitir o uso de passos genéricos e, consequentemente, a construção de planos de trabalho utilizáveis por mais do que uma reparação seria monumental na construção de um sistema relevante. Se possível estabelecer uma base de dados com esta informação, seria altamente eficaz o processo de definir orçamentos e realizar reparações.

Reparação Expresso com Plano de Trabalhos

Devido à redundância da existência de um plano de trabalhos para um serviço fixo e imediato, foi retirada a utilização de um plano de trabalhos à reparação expresso.

Stock do Material

Com a escolha de não ser implementada a manipulação do stock do material, a entidade **Quantidade** associada ao material já não se trata da sua quantidade em stock, mas sim a quantidade de materiais usados no passo de reparação a que estão associados. Sendo, por esta razão, os casos de uso, como 3.2.10, relativos a uma iteração futura do sistema.

Neste sentido, como resposta à necessidade de modelar o problema e com base na análise do enunciado proposto foram identificadas as principais entidades (2.2) que fazem parte do sistema, assim como os relacionamentos entre elas.

2.2 Entidades

2.2.1 Cliente

O cliente é a entidade que possui o(s) equipamento(s) que necessita(m) de intervenção técnica, cada cliente é identificado pelo seu Número de Identificação Fiscal (NIF) e pelas suas Formas de Contacto. Um cliente pode ser contactado por um colaborador através de um *SMS* ou *Email*.

2.2.2 Colaborador

O Colaborador é a entidade relativa a todos os elementos que interagem diretamente com o sistema. Numa forma geral, engloba todos os trabalhadores que populam o **Centro de Reparações** e, após autenticados, executam os serviços pedidos - Uma entidade abstrata que ramifica-se em sub-entidades(trabalhadores) mais específicas. Todos são identificados por um número de identificação, para permitir a sua organização.

Desde a inserção do equipamento a ser reparado no sistema até à sua eventual entrega ao cliente, o processo conta com participação do(s) **Funcionário(s) de Balcão** e do(s) **Colaboradore(s) Especializado(s)**.

Funcionário de Balcão

Corresponde à entidade responsável pelo início e o término de uma reparação - o registo inicial e entrega do equipamento ao cliente, respetivamente. Encontra-se associado aos equipamentos com o qual interage.

Colaborador Especializado

Esta entidade comprehende os colaboradores com competências técnicas especializadas, divergindo na sua autoridade com o Funcionário de Balcão. Estes possuem a capacidade de alterar o que considerarem necessário no Centro de Reparação.

Desta forma, foi definido os seguintes tipos:

Técnico Responsável por fazer o orçamento e realizar a reparação

Gestor Tem permissões de administração sobre todo o sistema.

2.2.3 Equipamento

Corresponde a uma entidade que representa um objeto físico que irá entrar no sistema com a perspetiva de obter um orçamento (calculado ou fixo) e, eventualmente, originar uma reparação. Esta entidade é identificada por um código de registo e marca. Optou-se por um identificador duplo - {marca, código de registo}, por forma a garantir que cada equipamento é facilmente identificado sem repetições. O código de registo é o número de série. Adicionalmente, para representar a dinâmico do equipamento dentro do centro de

reparações, tem um estado associado, que permite datar as várias iterações da sua reparação. Os estados possíveis para o equipamento:

- Em processo
- Pronto a Levantar
- Abandonado
- Entregue

2.2.4 Orçamento

O orçamento é a entidade associada a um plano de trabalhos estabelecido pelo técnico e ao prazo máximo de execução da reparação. De modo análogo ao equipamento, encontra-se parametrizado com estados durante a sua criação até ao seu envio e confirmação. Os estados do orçamento:

- Por Calcular
- Enviado
- Arquivado
- Aceite

2.2.5 Reparação

Representa um serviço disponibilizado no sistema. Mediante os cenários apresentados, foram definidos dois tipos: *Reparação expresso* e *Reparação normal*, possuindo, cada um, a sua dinâmica própria. Esta entidade caracteriza-se por um estado para o caracterizar no seu desenvolvimento. Os estados da reparação:

- Aguarda Reparação
- Em Reparação
- Reparada
- Cancelada

- Pagamento Recusado
- Pago

Reparação expresso

Define um tipo de reparação pré-definida. É um serviço fixo que, por sua vez, origina um custo e tempo de realização fixo.

Reparação normal

Corresponde a uma reparação personalizada que contém um plano de trabalhos próprio e único. Consequentemente, o preço e tempo varia entre cada uma.

2.2.6 Forma de Contacto

A forma de contacto permite uma comunicação com o cliente, sendo registada a data e hora e colaborador que a efetuou. Para tal, as opções existentes são o SMS e o Email.

2.2.7 Comunicação

Para definir uma interação realizada entre os colaboradores e o cliente, sendo datada e representada por uma mensagem.

2.2.8 Plano de Trabalhos

O plano de trabalhos é dividido em passos e sub-passos da reparação, onde cada um consome tempo e utiliza material. Esta definição permite obter o número total de horas de trabalho e o custo das peças utilizadas.

2.2.9 Passo de Reparação

Um passo de reparação é definido pelos seus sub-passos, o tempo e material necessário para a sua realização. São únicos e constituem os planos de trabalho das reparações programadas,

sendo a principal causa que o preço entre cada uma varia.

2.2.10 Material

O material é uma entidade fundamental tanto para a previsão do orçamento como para a própria reparação. Este representa a totalidade de materiais usados num passo. É identificado pela sua referência, está associado a um custo e tem a quantidade de material necessário. Adicionalmente, tem o nome dos materiais, para ser identificável.

2.2.11 Principais alterações nas Entidades

Com as mudanças feitas ao nível da definição do problema inicial, foram, consequentemente, cruciais retificações nas entidades desenvolvidas.

As principais alterações relativamente à primeira fase:

- A entidade cliente possui as suas formas de contacto.
- Colaboradores têm um número de identificação associado.
- Adicionados estados para parametrizar e datar os equipamentos, os orçamentos e as reparações.
- A reparação em si não tem um plano de trabalhos associado, sendo apenas a reparação programada.
- Consequentemente, a reparação expresso não tem um plano de trabalhos definido no sistema.
- O material já não se enquadra numa categoria de material.
- A quantidade do material não é relativa ao *stock*, mas sim ao número de materiais usados no passo.

2.3 Diagrama de Modelo de Domínio

No seguimento das várias alterações feitas ao diagrama apresentado na primeira fase do projeto, surge a comparação seguinte.

2.3.1 Modelo de Domínio Anterior

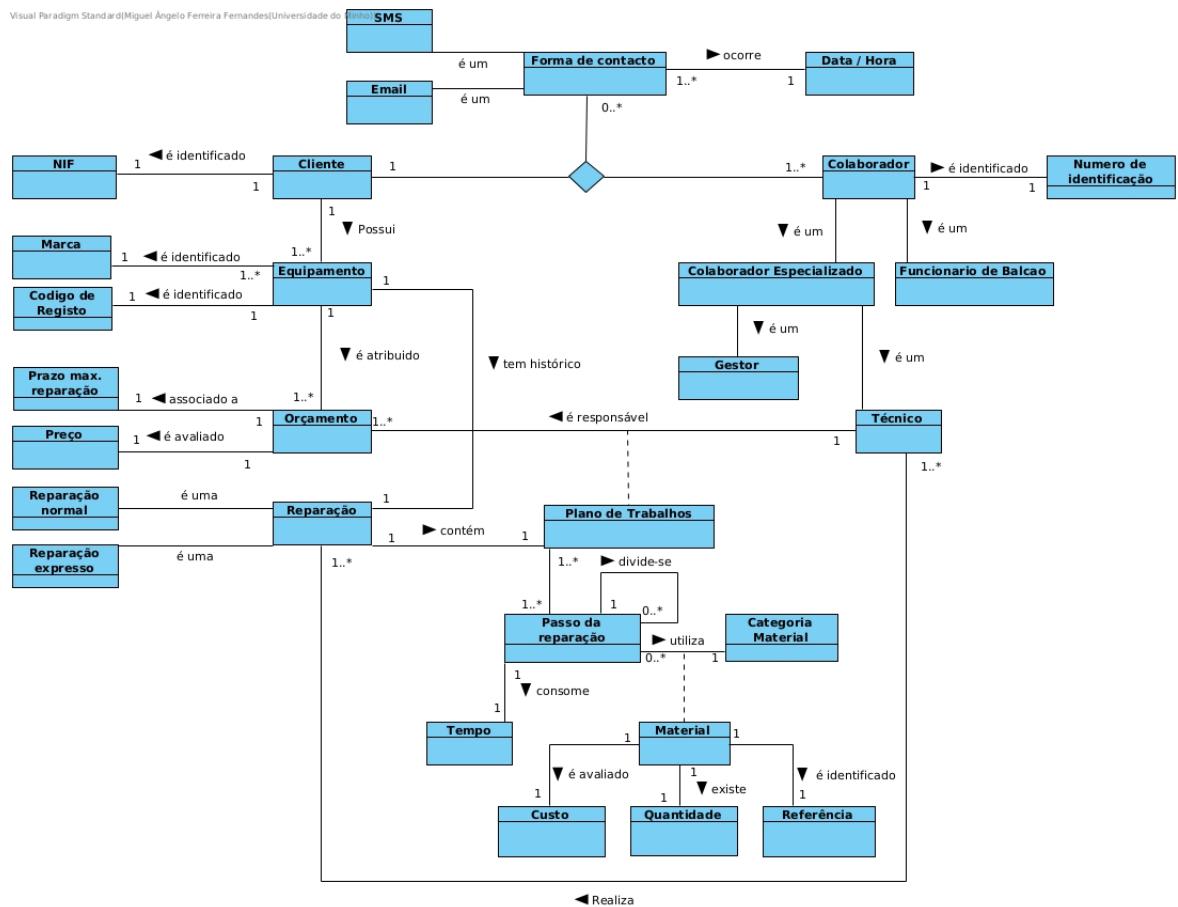


Figura 2.1: Modelo de Domínio

2.3.2 Modelo de Domínio Atual

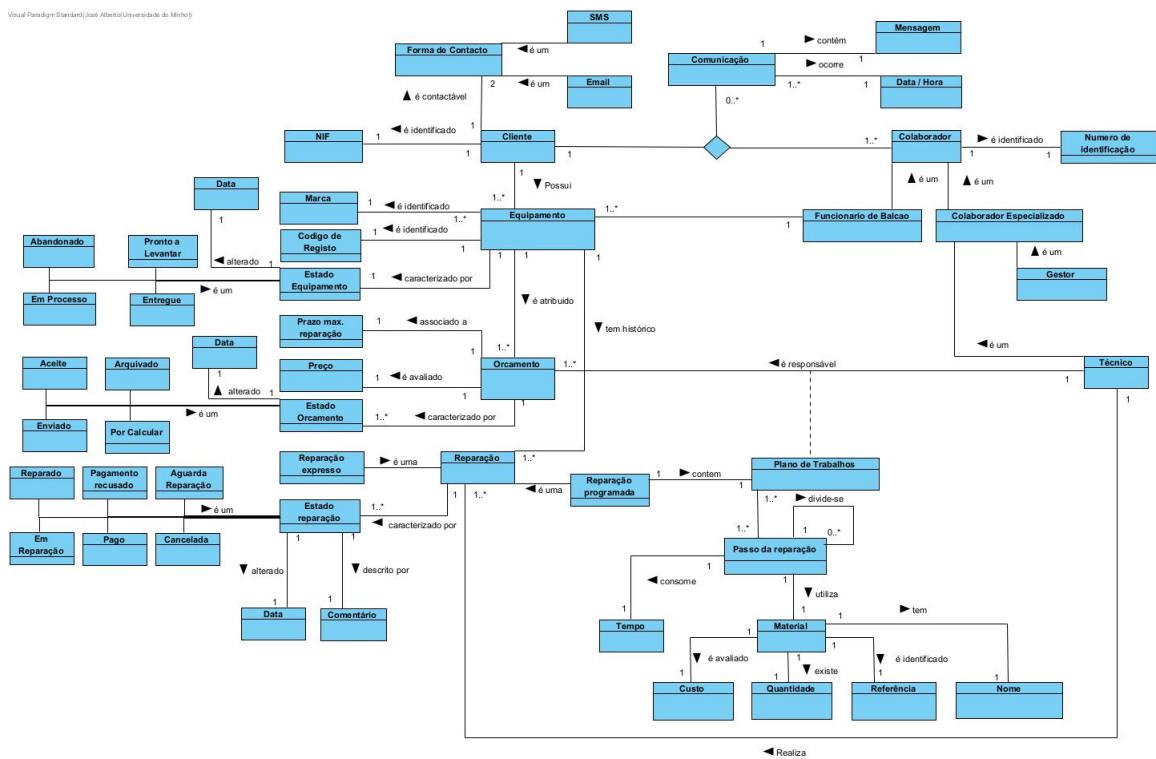


Figura 2.2: Modelo de Domínio

3 Modelação dos Requisitos Funcionais

A partir dos cenários (ver 6.1) identificados pela equipa docente, foram identificados os seguintes *use cases*:

- Criar reparação expresso - 3.2.1
- Pedir orçamento - 3.2.2
- Fazer orçamento - 3.2.3
- Criar passo da Reparação - 3.2.4
- Confirmar orçamento - 3.2.5
- Arquivar orçamento - 3.2.6
- Realizar reparação - 3.2.7
- Realizar passo de reparação - 3.2.8
- Adicionar material - 3.2.9
- Encomendar material - 3.2.10
- Receber material - 3.2.11
- Entregar equipamento - 3.2.12
- Dar baixo do equipamento - 3.2.13
- Pedir reparação expresso - 3.2.14
- Registar equipamento - 3.2.15
- Registar cliente - 3.2.16

- Registrar Colaborador - 3.2.17
- Autenticar Colaborador- 3.2.18
- Listar resumida do técnico - 3.2.19
- Listar detalhada do técnico - 3.2.20
- Listar funcionário do balcão - 3.2.21

3.1 Modelo de Use Cases

3.1.1 Principais Diferenças com a Primeira Fase

Em relação ao Diagrama de Use Cases desenvolvido, houve poucas alterações do que foi proposto na primeira fase do projeto. Previamente, os casos de Uso **Dar Baixa do Equipamento** (Ver 3.2.13) e **Arquivar Orçamento** (Ver 3.2.6) seriam efetuadas pelo ator *System Timer*. Contudo, após uma análise da aplicação e das dinâmicas expressas, entendeu-se como preferencial o **Gestor** efetuar estas funcionalidades, de modo a permitir visualizar os equipamentos que levaram baixa e os orçamentos que foram arquivados, no momento em que o Gestor escolhe. Foi retirado, então, o *System Timer* como ator no diagrama de Use Cases. Neste sentido, beneficia, também, a nível de implementação, não sendo necessário uma estratégia de criar o *System Timer* dentro do sistema.

É necessário demarcar a presença dos casos de uso:

- Adicionar Material
- Encomendar Material
- Receber Material

Apesar da sua implementação não ter sido realizada, no estado atual da aplicação, são apresentados para representar funcionalidades que seriam implementadas em iterações seguintes do sistema. Estes casos de uso lidam com a manipulação de *stocks* e, no tipo de projeto a realizar, foram caracterizadas como não necessárias apresentar nesta iteração do sistema. Adicionalmente, a exclusão do desenvolvimento destas funcionalidades facilitou a implementação do sistema.

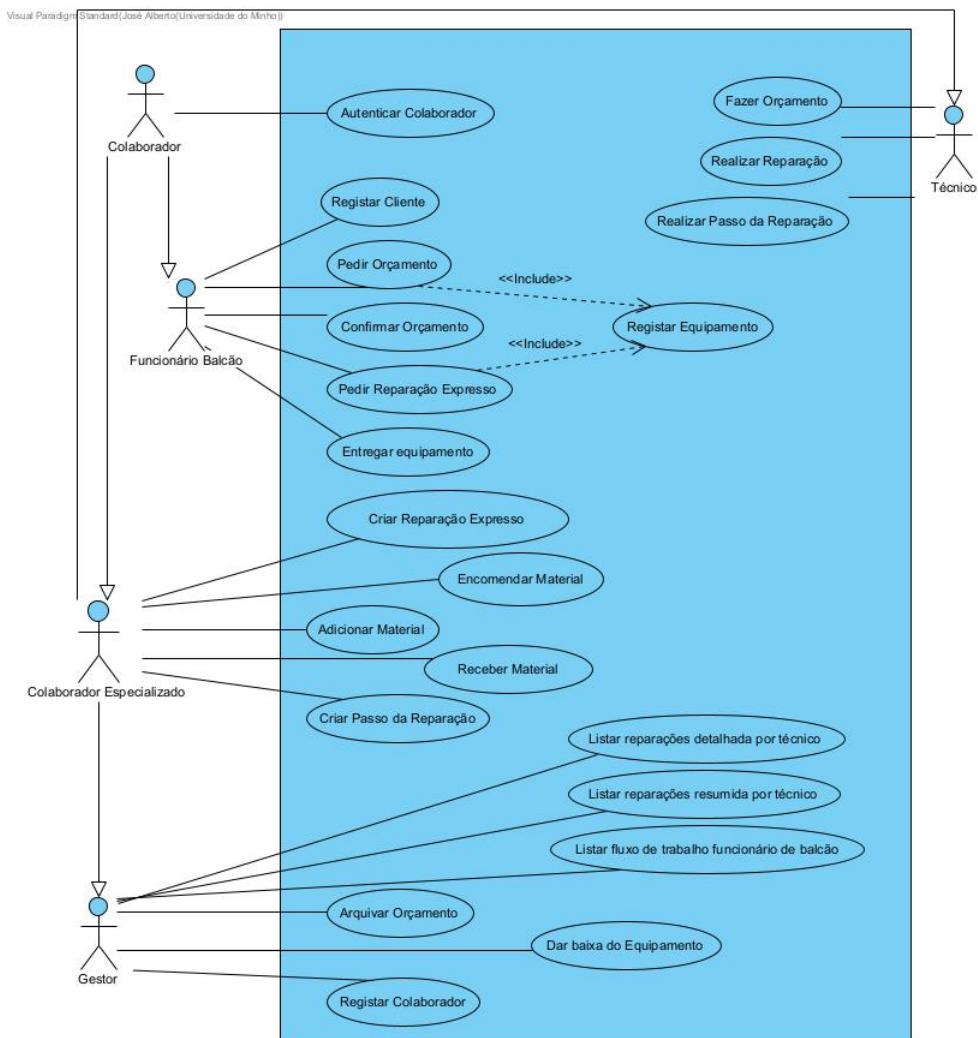


Figura 3.1: Diagrama de *Use Case*

3.2 Especificação Use Cases

3.2.1

Registo de uma nova reparação expresso no sistema, incluindo o material utilizado, tempo consumido e preço. Devido à reparação expresso não possuir um plano de trabalhos definido no sistema, o Colaborador especializado apenas insere o nome, preço e tempo estimados.

Use Case Criar Reparação Expresso

Cenários 2

Pré-condição Colaborador especializado autenticado

Pós-condição Reparação expresso fica registada no Sistema

Fluxo Normal

1. Colaborador especializado insere nome da reparação a adicionar
2. Sistema regista nome da reparação
3. Colaborador especializado insere dados da reparacao
4. Sistema regista reparação expresso

Fluxo de Exceção 1 (passo 2) [Nome da reparação já existe]

- 2.1 Sistema comunica que nome da reparação já existe

Use Case	Fluxo	Responsabilidade	API	Subsistema
Colaborador especializado inserir nome da reparação	UI			
Sistema verifica nome da reparação		verifica se o nome já existe	existeRepXpresso(nome : String) : Boolean	SSReparacoes
Colaborador especializado inserir dados da reparacao	UI			
Sistema regista reparação expresso		regista expresso	registarRepXpresso(nome : String, tempo : Integer, preco : Double)	SSReparacoes
Exceção	1		Nome da reparação já existe	
Sistema comunica que nome da reparação já existe	UI			

Tabela 3.1: Análise Use Case - Criar Reparação Expresso (ver 3.2.1)

3.2.2

Resgisto do pedido de orçamento de um equipamento através do seu identificador.

A única alteração realizada em relação à primeira fase foi a de acrescentar a responsabilidade ao sistema de registar a receção de um equipamento, tal como no *Use Case* "Realizar passo da reparação".

Use Case Pedir orçamento

Cenários 1 e 2

Pré-condição Funcionário autenticado e cliente registado

Pós-condição Pedido de orçamento adicionado

Fluxo Normal

1. Funcionário insere identificador do cliente
2. Sistema seleciona cliente
3. Funcionário insere identificador do equipamento
4. Sistema verifica identificador do equipamento
5. Funcionário insere descrição da avaria regista pedido de orçamento
6. Sistema adiciona pedido de orçamento
7. Sistema regista receção do equipamento

Fluxo Alternativo 1 (passo 4) [Equipamento não registado]

- 2.1 << *include* >> Registar equipamento
- 2.2 Regressa ao passo 3

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário insere identificador do cliente	UI			
Sistema seleciona cliente		procura cliente	getCliente(CliID : String) : Cliente	SubClientes
Funcionário insere identificador do equipamento	UI			
Sistema verifica identificador do equipamento		verifica se o equipamento existe	existeEquipamento(codR : String, marca: String) : Boolean	SubClientes
Funcionário regista pedido de orçamento	UI			
Sistema adiciona pedido de orçamento	Sistema	regista pedido de orçamento e receção do equipamento	registarOrcamento(nif : String, equipID : String, descr: String, fnld : String)	SubReparacoes
Alternativo	1			
$\ll include \gg$ Registar equipamento			Equipamento não registado	
Regressa ao passo 3				

Tabela 3.2: Análise Use Case - Pedir orçamento (ver 3.2.2)

3.2.3

Registo de um orçamento relativo a um pedido de orçamento, através do seu identificador. O técnico constrói o plano de trabalhos e, quando finalizado, o sistema envia orçamento ao cliente, registando data e hora.

Use Case Fazer orçamento

Cenários 3

Pré-condição Técnico autenticado e existe pedidos de orçamento

Pós-condição Orçamento gerado, registado e comunicado ao cliente

Fluxo Normal

1. Técnico solicita pedido de orçamento mais antigo
2. Sistema devolve pedido de orçamento mais antigo
3. Técnico utiliza identificador para levantar equipamento
4. Técnico define plano de trabalhos para a reparação
5. Sistema regista plano de trabalhos para a reparação
6. Sistema gera orçamento e regista-o
7. Técnico regista contacto com o cliente
8. Sistema regista data e hora de comunicação

Fluxo de Exceção 1 (passo 4) [Equipamento não pode ser reparado]

- 4.1 Técnico regista que não é possível fazer reparação
- 4.2 Sistema comunica o cliente

Use Case	Fluxo	Responsabilidade	API	Subsistema
Técnico solicita pedido de orçamento mais antigo	UI			
Sistema devolve pedido de orçamento mais antigo		Calcula o pedido de orçamento de orçamento há mais tempo no sistema	getOrcamentoMaisAntigo(): Orcamento	SSReparacoes
Técnico define plano de trabalhos para a reparação	UI			
Sistema regista plano de trabalhos para a reparação		Criar passos e inserir no plano de trabalhos do Orçamento	criarPasso(String orclD, String nomePasso, String mat, Integer tempo, Integer qMat, Double custoMat)	SSReparacoes
Sistema gera orçamento e regista-o		Cria um documento com o resumo do orçamento (plano de trabalho + preço)	generateResume()	SSReparacoes
Técnico regista contacto com o cliente	UI			
Sistema regista data e hora de comunicação		comunicarErro(orcid: String, msg: String, tecId: Tecnico)		SSReparacoes
Exceção	1		Equipamento não pode ser reparado	
Técnico regista que não é possível fazer reparação	UI			
Sistema comunica o cliente		envia mensagem ao cliente que há erros	comunicarErro(orcid: String, msg: String, tecId: Tecnico)	SSReparacoes

Tabela 3.3: Análise Use Case - Fazer orçamento (ver 3.2.3)

3.2.4

Registo no sistema de um novo passo, introduzindo o tempo e peças necessárias para a sua concretização. Como referido anteriormente, a **categoria de material** e a manipulação do *stock* não é feita nesta iteração do projeto, contudo apresentam-se aqui para demonstrar como se incluiam, caso fossem futuramente implementadas. Neste sentido, o terceiro, quarto e quinto passos do fluxo normal e os fluxos alternativos 2 e 3 representam funcionalidades que não estão presentes de momento.

Use Case Criar passo de reparação

Cenários 3

Pré-condição Colaborador especializado autenticado

Pós-condição Passo de reparação é adicionado ao Sistema

Fluxo Normal

1. Colaborador especializado insere nome do passo da reparação
2. Sistema verifica nome do passo da reparação
3. Colaborador especializado insere dados do passo da reparação
4. Colaborador especializado insere categoria do material
5. Sistema verifica categoria do material
6. Sistema regista passo de reparação

Fluxo de Exceção 1 (passo 2) [Nome já existe]

- 2.1 Sistema comunica que nome do passo da reparação já existe

Fluxo Alternativo 2 (passo 4) [Material não registado]

- 4.1 << include >> Adicionar Material
- 4.2 Regressa ao passo 5

Fluxo Alternativo 3 (passo 5) [Categoria não existe]

- 5.1 Sistema comunica que categoria não existe

5.2 Colaborador especializado insere nome da categoria

5.3 Sistema regista nova categoria

5.4 Regressa ao passo 6

Use Case	Fluxo	Responsabilidade	API	Subsistema
Colaborador especializado inserir nome do passo da reparação	UI			
Sistema verifica nome do passo da reparação	UI	verifica se o nome do passo já existe	existePasso(nomePasso : String) : boolean	SSReparacoes
Colaborador especializado inserir dados do passo da reparação	UI			
Colaborador especializado inserir categoria do material	UI			
Sistema verifica categoria do material		verifica se categoria existe	existeCategoria(nomeCategoria):boolean	SSReparacoes
Sistema regista passo de reparação		regista passo de reparação com os dados passados	criarPasso(String orclID, String nomePasso, String mat, Integer tempo, Integer qMat, Double custoMat)	SSReparacoes
Exceção	1		Nome já existe	
Sistema comunica que nome do passo da reparação já existe	UI			
Alternativo	2		Material não registado	
<< include >> Adicionar Material				
Regressa ao passo 5				
Alternativo	3		Categoria não existe	
Sistema comunica que categoria não existe	UI			
rowcoloryellow Colaborador especializado insere nome da categoria	UI			
Sistema regista nova categoria		registra categoria	registraCategoria(nomeCategoria : String)	SSReparacoes
Regressa ao passo 6				

3.2.5

Registo da confirmação do orçamento, aquando da aceitação do cliente.

Em termos estruturais, o use case **Confirmar Orcamento** encontra-se semelhante ao seu estado anterior. Foram adicionados alguns passos ao fluxo normal como o quinto e sexto, pois o método está preparado para criar uma reparação e a associar a um técnico, atualizando a sua agenda. Foram alterados os métodos na análise de requisitos, pelo que a lógica toda é tratada pelo método *AlteraEstadoOrc*.

Use Case Confirmar orçamento

Cenários 1 e 3

Pré-condição Funcionário autenticado e orçamento concluído

Pós-condição Adicionado pedido de reparação ao sistema

Fluxo Normal

1. Funcionário insere o identificador do orçamento
2. Sistema devolve orçamento
3. Funcionário regista orçamento como aceite
4. Sistema atualiza orçamento para reparação
5. Sistema associa reparação a um Técnico
6. Sistema atualiza agenda do Técnico
7. Sistema insere reparação à lista de equipamentos por reparar

Fluxo de exceção 1 (passo 3) [Cliente recusa reparação]

- 3.1 Funcionário regista orçamento como recusado
- 3.2 Sistema adiciona orçamento ao arquivo
- 3.3 Sistema coloca equipamento pronto a levantar

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário pede orçamento ao sistema	UI			
Sistema devolve orçamento		obtém o orçamento	getOrcamento(ref: String): Orcamento	SSReparacoes
Funcionário registra orçamento como aceite	UI			
Sistema atualiza orçamento para reparação Sistema associa reparação a um Técnico		Adiciona a reparação à lista de reparações por efetuar	addReparacao(orc : Orcamento, tec: Tecnico)	SSReparacoes
Sistema atualiza agenda do Técnico		Adiciona um evento à agenda do Técnico	addEventoAgenda(String teclId, Integer tempo, String detalhes)	SSColaboradores
Sistema insere pedido de reparação à lista de equipamentos por reparar		Criar reparação e inseri-la no sistema, associando-a a um técnico	alteraEstadoOrc(orcID: OrcamentoEstado)	SSReparacoes
Funcionário registra orçamento como recusado	UI			
Exceção	1		Cliente recusa reparação	
Sistema adiciona orçamento ao arquivo Sistema coloca equipamento disponível para entrega		muda o estado do orçamento para arquivado e o do equipamento associado para Pronto a Levar	alteraEstadoOrc(orcID: OrcamentoEstado)	String, SSReparacoes

Tabela 3.5: Análise Use Case - Confirmar Orcamento (ver 3.2.5)

3.2.6

Arquivamento dos orçamentos que não obtiveram resposta por parte do cliente há mais de 30 dias. A decisão do arquivo é feita pelo Gestor.

Em relação ao fluxo do caso de uso apresentado na primeira fase, o presente diferencia-se no ator que efetua o arquivo. Previamente uma tarefa do *System Timer*, com a remoção deste, foi necessária a mudança para um ator novo, sendo o mais apropriado o Gestor. Como este possuí privilégios relativamente aos outros colaboradores, é considerado o mais apropriado a efetuar a decisão. Apesar disto, os métodos definidos para a análise de requisitos da segunda fase permanecem os mesmos.

Use Case Arquivar orçamento

Cenários 1

Pré-condição Gestor autenticado

Pós-condição Orçamentos com mais de 30 dias arquivados

Fluxo Normal

1. Gestor pede para arquivar os orçamentos que ja passaram 30 dias desde que foram enviados
2. Sistema filtra todos os orçamentos enviados
3. Sistema filtra orçamentos por data, ficando com os superiores a 30 dias
4. Sistema adiciona orçamentos ao arquivo e apresenta-os

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor pede para arquivar os orçamentos que já passaram 30 dias desde que foram enviados	UI			
Sistema filtra todos os orçamentos enviados Sistema filtra orçamentos por data, ficando com os superiores a 30 dias	filtrar orçamentos com mais de 30 dias	filterOrcamentos(p : Orcamento<Orcamento> enviado() & passouPrazo()): List<Orcamento>		SSReparacoes
Sistema adiciona orçamentos ao arquivo	mudar o estado do orçamento	arquivaOrcamentos():List<Orcamento>		SSReparações

Tabela 3.6: Identidade do Projeto.

3.2.7

Reparação de um equipamento, realizada por um técnico, tendo em conta o plano de trabalhos.

Como a adição das agendas, já não é viável pedir a reparação mais urgente, uma vez que todas as reparações já foram distribuídas uniformemente pelos diferentes técnicos. Outra modificação efetuada neste *Use Case* foi a alteração do estado da reparação aquando do início da reparação para "emReparacao".

Na primeira fase estava estabelecido que quando uma reparação era interrompida, esta era adicionada à lista de reparações pendentes, porém como referido anteriormente esta lista deixou de existir com o uso das agendas e a solução encontrada foi alterar o estado da reparação para "aguardaReparacao". Apesar de se verificar esta alteração ao nível do *Use Case*, o mesmo não acontece ao nível da implementação que não foi estabelecido nenhum mecanismo capaz de lidar com interrupções.

Use Case Realizar reparação

Cenários 4

Pré-condição Técnico autenticado e equipamentos para reparar

Pós-condição Equipamento reparado

Fluxo Normal

1. Técnico consulta a sua agenda e indica o identificador da reparação a ser efetuada
2. Sistema devolve reparação
3. Sistema altera estado da reparação para em reparacao
4. Técnico pede o plano de trabalhos da reparação
5. Técnico indica os passos da reparação concretizados
6. Sistema regista passos da reparação como realizados
7. Sistema regista a conclusão da reparação
8. Sistema muda estado da reparação para reparada
9. Técnico coloca equipamento disponível para levantamento

10. Sistema muda estado de equipamento para pronto para levantamento
11. Sistema regista data e hora

Fluxo de Exceção 1 (passo 7) [falta de tempo]

- 3.1 Técnico informa motivo de interrupção
- 3.2 Sistema altera estado da reparação para aguarda reparacao

Fluxo de Exceção 2 (passo 7) [falta de peças]

- 3.1 Técnico informa motivo de interrupção
- 3.2 Sistema altera estado da reparação para aguarda reparacao

Fluxo Alternativo 3 (passo 7) [reparação excede valor orçamentado]

- 3.1 Técnico insere contacto com cliente
- 3.2 Sistema regista data, hora e técnico associado
- 3.3 Regressa ao passo 3

Fluxo de Exceção 4 (passo 3.2, fluxo 2) [cliente recusa continuidade da reparação]

- 3.2.1 Sistema coloca reparação como cancelada
- 3.2.2 Técnico repõe o equipamento
- 3.2.3 Técnico regista as horas gastas
- 3.2.4 Técnico coloca equipamento disponível para levantamento
- 3.2.5 Sistema muda estado de equipamento para pronto para levantamento
- 3.2.6 Sistema muda estado da reparação para cancelada
- 3.2.7 Sistema regista data e hora

Use Case		Fluxo	Responsabilidade	API	Subsistema
Técnico consulta a sua agenda e indica o identificador da reparação a ser efetuada	UI				
Sistema devolve reparação		seleciona a reparação indicada	getReparacao(repID : String) : Reparacao	SubReparacoes	
Sistema altera estado da reparação para emReparacao		muda estado de reparação para emReparacao	alterarEstadoRep(repID : String , estado : ReparacaoEstado)	SubReparacoes	
Técnico pede o plano de trabalhos da reparação	UI				
Técnico indica os passos da reparação concretizados	UI				
Sistema regista passos da reparação como realizados		regista os passos da reparação realizados	registarPassosRep(repID : String, passosRealizados : List<PassoReparacao>)	SubReparacoes	
Sistema regista a conclusão da reparação	Sistema	regista reparação como concluida	alterarEstadoRep(repID : String, estado : EstadoReparacao)	SubReparacoes	
Técnico coloca equipamento disponível para levantamento	UI				
Sistema muda estado de equipamento para pronto para levantamento	Sistema	muda estado de equipamento para pronto para levantamento e regista hora	alteraEstadoEq(eqIID : String, state : EstadoEquipamento)	SubClientes	
Exceção	1			falta de tempo	
Técnico informa motivo de interrupção	UI				
Sistema altera estado da reparação para aguardaReparacao		muda estado de reparação para aguardaReparacao	alterarEstadoRep(repID : String , estado : ReparacaoEstado)	SubReparacoes	
Exceção	2			falta de peças	
Técnico informa motivo de interrupção	UI				

3.2.8

Progressão de uma reparação, através da realização do passo. É atualizado o *stock* do material gasto e registado do tempo efetivamente consumido.

De maneira a não aumentar a complexidade do trabalho não foram considerados stocks de materiais, apesar de estes serem referidos neste *Use Case*. Assim quando se realiza o passo não é necessário decrementar o stock em função da quantidade de material gasto. Como referido anteriormente, a adição da agenda permitiu que as reparações ficassem logo atribuídas a um técnico no momento em que foram criadas. Assim já não é possível uma reparação ser realizada por mais que um técnico e ao contrário do que foi estabelecido na primeira fase já não é necessário identificar o autor de cada passo.

Use Case

Cenários 4

Pré-condição Técnico autenticado e material necessário disponível

Pós-condição Passo adicionado à lista de passos realizados

Fluxo Normal

1. Técnico indica a identificação da reparação a ser efetuada
2. Sistema seleciona a reparação
3. Sistema seleciona o passo de reparação atual
4. Técnico indica referência do material gasto
5. Sistema altera *stock* do material gasto
6. Técnico indica tempo e custo efetivamente gasto
7. Sistema regista tempo e custo efetivamente gasto
8. Sistema adiciona passo à lista de passos realizados

Use Case	Fluxo	Responsabilidade	API	Subsistema
Técnico indica a identificação da reparação a ser efetuada	UI			
Sistema seleciona a reparação		Sistema seleciona a reparação a partir do seu ID	getRep(repID: String): Reparacao	SubReparacoes
Sistema seleciona o passo de reparação atual		Sistema seleciona o passo de reparação a partir do seu ID	getPassoAtual(): PassoReparacao	SubReparacoes
Técnico indica referência do material gasto	UI			
Sistema altera stock do material gasto		Sistema decremente o stock do material gasto	consumeMaterial(mat: Material)	SubReparacoes
Técnico indica tempo e custo efectivamente gasto	UI			
Sistema regista tempo e custo efectivamente gasto e Técnico adiciona passo à lista de passos realizados		Sistema regista os valores do passo após ter sido concretizado	registraPassoRealizado(tempo : Integer, custo : Double)	In-SubReparacoes

Tabela 3.8: Análise Use Case - Realizar passo da reparação (Ver 3.2.8)

3.2.9

Adicionar um material ao sistema, utilizando os respetivos dados.

Use Case Adicionar material

Cenários 3

Pré-condição Colaborador especializado autenticado

Pós-condição Material é adicionado ao Sistema

Fluxo Normal

1. Colaborador especializado insere referência do material
2. Sistema verifica código de registo
3. Colaborador especializado insere custo do material
4. Sistema adiciona material ao Sistema

Fluxo de Exceção 1 (passo 2) [Código de registo já no sistema]

- 2.1 Sistema informa que material já está registado

Use Case	Fluxo	Responsabilidade	API	Subsistema
Colaborador especializado inserir referência do material	UI			
Sistema verifica código de registro		verifica se código já existe	existeMaterial(ref: String): boolean	SubReparacoes
Colaborador especializado inserir custo do material	UI			
Sistema adiciona material ao Sistema		adiciona material	addMaterial(ref: String, qnt: Int, nome: String, custo: Float, categorias: List<String>)	SubReparacoes
Sistema informa que material já está registado	UI			

Tabela 3.9: Identidade do Projeto.

3.2.10

Encomenda do material que se encontra com o seu *stock* a 0 e que é necessário para a concretização de algum passo.

Use Case Encomendar material

Cenários 3 e 4

Pré-condição Colaborador especializado autenticado e reparações pendentes

Pós-condição Sistema regista nova encomenda

Fluxo Normal

1. Colaborador especializado pede material em falta para realizar reparações
2. Sistema calcula o material necessário para concluir os passos das reparações pendentes
3. Sistema calcula o material disponível
4. Sistema calcula o material necessário e sem *stock* disponível
5. Sistema regista uma nova encomenda com o material não disponível mas necessário

Fluxo de Exceção 1 (passo 4) [material necessário está em stock]

- 4.1 Sistema comunica que não é necessário encomendar material

Use Case	Fluxo	Responsabilidade	API	Subsistema
Colaborador especializado pede material em falta para realizar reparações	UI			
Sistema calcula o material necessário para concluir os passos das reparações pendentes	Sistema obtém o material necessário para concluir todas as reparações pendentes	getMaterial(): List<Material>		SubReparacoes
Sistema calcula o material disponível	Sistema calcula o material disponível	getMaterialDisponivel(): List<Material>		SubReparacoes
Sistema calcula o material necessário e sem stock disponível	Sistema calcula o material que não tem stock e é necessário para as reparações	getMaterialSStock(): List<Material>		SubReparacoes
Sistema regista uma nova encomenda com o material não disponível mas necessário	Registar uma nova encomenda com o material requerido para as reparações	addEncomenda(List<Material>)		SubReparacoes
Exceção	1	material necessário está em stock		
Sistema comunica que não é necessário encomendar material	UI			

Tabela 3.10: Análise Use Case - Encomendar Material (Ver 3.2.10)

3.2.11

Registo da nova quantidade de material recebida.

Use Case Receber material

Cenários 4

Pré-condição Colaborador Especializado autenticado

Pós-condição Material com nova quantidade disponível

Fluxo Normal

1. Colaborador Especializado indica referência do material recebido
2. Colaborador Especializado indica quantidade do material recebido
3. Sistema regista nova quantidade do material disponível

3.2.12

Representa o ato de entregar equipamento ao cliente. Compreende a mudança de estado do equipamento para entregue e, se aplicável, o registo do pagamento do equipamento.

Use Case Entregar equipamento

Cenários 1, 2 e 4

Pré-condição Funcionário autenticado e equipamento disponível para entrega

Pós-condição Equipamento entregue e pago

Fluxo Normal

1. Funcionário insere identificador do equipamento
2. Sistema devolve preço da reparação do equipamento
3. Funcionário regista a reparação como paga
4. Sistema regista a reparação como paga

5. Funcionário define equipamento como entregue
6. Sistema altera estado do equipamento para entregue

Fluxo de Exceção 1 (passo 2) [cliente recusou orçamento]

- 2.1 Funcionário define equipamento como entregue
- 2.2 Sistema altera estado do equipamento para entregue

Fluxo de Exceção 2 (passo 3) [cliente não paga]

- 3.1 Funcionário regista comentário que cliente não pagou
- 3.2 Sistema regista comentário
- 3.3 Sistema regista data e hora

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário insere identificador do equipamento	UI			
Sistema devolve preço da reparação do equipamento		Sistema calcula o preço da reparação do equipamento	calcularPrecoRep(repID:String): Double	SubReparacoes
Funcionário regista reparação como pago	UI			
Sistema regista a reparação como paga		Sistema altera o estado da reparação para paga	alterarEstadoRep(repID: String, estado: EstadoReparacao)	SubReparacoes
Funcionário define equipamento como entregue	UI			
Sistema altera estado do equipamento para entregue		Sistema altera o estado do equipamento para entregue	alteraEstadoEq(eqID: String, state: EstadoEquipamento)	SubReparacoes
Exceção	1		cliente recusou orçamento	
Funcionário define equipamento como entregue	UI			
Sistema altera estado do equipamento para entregue		Sistema altera estado do equipamento para entregue	alteraEstadoEq(eqID: String, state: EstadoEquipamento)	SubReparacoes
Exceção	2		cliente não paga	
Funcionário regista comentário que cliente não pagou	UI			
Sistema regista comentário		Sistema regista um comentário na reparação e a data/hora que foi feito	alterarEstadoRep(repID: String, estado: EstadoReparacao, comentario: String)	SubReparacoes
Sistema regista data e hora				

Tabela 3.11: Análise Use Case - Entregar equipamento (Ver 3.2.12)

3.2.13

Arquivamento do equipamento pronto para levantar há 90 dias. Análogo ao caso de uso **Arquivar Orçamentos**, este, também, substitui o ator *System Timer* pelo Gestor, dado a ser considerado mais apropriado para tomar esta decisão.

Em termos de métodos, a lógica toda é tratada a partir do método *darBaixaEquipamento*.

Use Case Dar baixa do equipamento

Cenários 1

Pré-condição Gestor autenticado

Pós-condição Equipamento pronto para levantar há 90 dias é adicionado à lista de equipamentos abandonados

Fluxo Normal

1. Gestor informa que é uma hora definida
2. Sistema calcula todos os equipamentos prontos a levantar
3. Sistema filtra equipamentos prontos a levantar há mais de 90 dias
4. Sistema adiciona equipamentos à lista de equipamentos abandonados

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor informa que é uma hora definida	UI			
Sistema calcula todos os equipamentos prontos a levantar Sistema filtra equipamentos prontos a levantar há mais de 90 dias Sistema adiciona equipamentos à lista de equipamentos abandonados	Sistema adiciona os equipamentos prontos à 90 dias como abandonados		darBaixaEquipamentos():List<Equipamento>	SSCClientes

Tabela 3.12: Análise Use Case - Dar Baixa Equipamento (Ver 3.2.13)

3.2.14

Registo do pedido de um serviço expresso.

Neste *Use Case* foram acrescentadas duas responsabilidades a mais ao Sistema, uma delas é que ao verificar a disponibilidade tem de também devolver qual o técnico que é capaz de realizar a reparação de imediato. A outra responsabilidade advém da necessidade de listar o trabalho do funcionário de balcão sendo assim necessário registar a receção de um equipamento.

Use Case Pedir reparação expresso

Cenários 2

Pré-condição Funcionário autenticado e cliente registado

Pós-condição Equipamento adicionado à lista de equipamentos por reparar

Fluxo Normal

1. Funcionário insere tipo da reparação expresso
2. Sistema verifica se existe a reparação expresso
3. Sistema verifica disponibilidade
4. Sistema devolve técnico com disponibilidade
5. Funcionário insere identificador do cliente
6. Sistema seleciona cliente
7. Funcionário insere identificador do equipamento
8. Sistema seleciona equipamento
9. Funcionário regista pedido de reparação
10. Sistema adiciona pedido
11. Sistema regista receção do equipamento

Fluxo de Exceção 1 (passo 3) [Não existe disponibilidade de tempo]

- 1.1 Sistema comunica que não existe disponibilidade

1.2 Funcionário recusa serviço

Fluxo Alternativo 2 (passo 8) [Equipamento não registado]

3.1 $\ll include \gg$ Registar equipamento

3.2 Regressa ao passo 6

Fluxo de Exceção 3 (passo 2) [tipo de reparação não é expresso]

4.1 Sistema comunica que reparação não é expresso

4.2 Funcionário recusa serviço

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário insere tipo da reparaçao expresso	UI			
Sistema verifica se existe a reparaçao expresso		verifica se existe a reparaçao expresso	existRepXpresso(nomeRepXpresso : String) : Boolean	SubReparacoes
Sistema verifica disponibilidade Sistema devolve técnico com disponibilidade com disponibilidade		verifica se existe disponibilidade e devolve técnico com disponibilidade	existDisponibilidade(duracao : Integer) : String	SubReparacoes
Funcionário insere identificador do cliente	UI			
Sistema seleciona cliente		seleciona o cliente	getCliente(CliID : String) : Cliente	SubClientes
Funcionário insere identificador do equipamento	UI			
Sistema seleciona equipamento		seleciona o equipamento	getEquipamento(EquipID : String) : Equipamento	SubClientes
Funcionário regista pedido de reparação	UI			
Sistema adiciona pedido Sistema regista receção do equipamento		adiciona pedido de reparação expresso ao sistema e regista receção do equipamento	addRepExpresso(equipID : String, nomeRepXpresso : String, funcId : String)	SubReparacoes
Exceção	1		Não existe disponibilidade de tempo	
Sistema comunica que não existe disponibilidade	UI			
Funcionário recusa serviço	UI			
Alternativo	2		Equipamento não registado	
<< include >> Registar equipamento				
Regressa ao passo 3				
Exceção	3		tipo de reparação não é expresso	
Sistema comunica que reparação não é expresso	UI			
Funcionário recusa serviço	UI			

3.2.15

Registo de um equipamento no sistema, utilizando os respetivos dados.

Use Case Registar equipamento

Cenários 1

Pré-condição Funcionário autenticado e cliente existente

Pós-condição Equipamento inserido no sistema

Fluxo Normal

1. Funcionário insere código de registo e marca do equipamento
2. Sistema verifica identificador {código de registo, marca} do equipamento
3. Funcionário insere identificador do cliente
4. Sistema regista os dados do equipamento

Fluxo de Exceção 1 (passo 2) [Identificador {código de registo, marca} já existe]

- 2.1 Sistema comunica que identificador já está registado

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário insere código de registo e marca do equipamento	UI			
Sistema verifica identificador {código de registo, marca} do equipamento		verifica se o equipamento já existe	existeEquipamento(codR : String, marca: String) : Boolean	SubClientes
Funcionário insere identificador do cliente	UI			
Sistema regista os dados do equipamento		regista equipamento	registaEquipamento(codR : String, marca: String, nif : String)	SubClientes
Exceção 1				
Sistema comunica que identificador já está registado	UI			

Tabela 3.14: Análise Use Case - Registar equipamento (ver 3.2.15)

3.2.16

Registo de um cliente no sistema, utilizando os respetivos dados.

Use Case Registrar cliente

Cenários 1 e 2

Pré-condição Colaborador autenticado

Pós-condição Novo cliente fica registado no sistema

Fluxo Normal

1. Funcionário insere identificador (NIF) do cliente
2. Sistema verifica identificador
3. Funcionário insere dados do cliente
4. Sistema regista o cliente

Fluxo de Exceção 1 (passo 2) [identificador já existe]

- 2.1 Sistema comunica que identificador já está registado

Use Case	Fluxo	Responsabilidade	API	Subsistema
Funcionário insere identificador (NIF) do cliente	UI			
Sistema verifica identificador do cliente		verifica se cliente existe	existeCliente(CliID : String) : Boolean	SubUtilizadores
Funcionário insere dados do cliente	UI			
Sistema regista o cliente		regista cliente	registaCliente(CliID : String, numero : String, email : String)	SubUtilizadores
Exceção	1		identificador já existe	
Sistema comunica que identificador já está registado	UI			

Tabela 3.15: Análise Use Case - Registar Cliente (ver 3.2.16)

3.2.17

Registo de um novo colaborador no sistema pelo Gestor, sendo-lhe associado um número de identificação.

Use Case Registrar Colaborador

Cenários 1, 2, 3, 4 e 5

Pré-condição Gestor autenticado

Pós-condição Novo colaborador registado

Fluxo Normal

1. Gestor insere nome e função do Colaborador
2. Sistema calcula número de identificação
3. Sistema regista Colaborador

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor insere nome e função do Colaborador	UI			
Sistema calcula número de identificação		calcula número de identificação do novo colaborador	GetCurrentID() : Integer	SubColaboradores
Sistema regista Colaborador		regista colaborador	registarColaborador(nome : String, tipo : Class<?>) : String	SubColaboradores

Tabela 3.16: Análise Use Case - Registar Colaborador (ver 3.2.17)

3.2.18

Autenticação de um colaborador no sistema, validando o seu número de identificação.

Use Case Autenticar Colaborador

Cenários 1, 2, 3, 4 e 5

Pré-condição Colaborador registrado

Pós-condição Colaborador autenticado

Fluxo Normal

1. Colaborador insere número de identificação
2. Sistema valida número de identificação

Use Case	Fluxo	Responsabilidade	API	Subsistema
Colaborador insere número de identificação	UI			
Sistema valida número de identificação		Verifica se o número de identificação existe e esta relacionado a um colaborador	existeColaborador(id: String): Boolean	SS Colaboradores

Tabela 3.17: Análise Use Case - Autenticar Colaborador (ver 3.2.18)

3.2.19

Extração de lista resumida de reparações realizadas, ordenadas por técnico e filtrado por data.

Use Case Listar reparações resumida por técnico

Cenários 5

Pré-condição Gestor autenticado e reparações efetuadas

Pós-condição Listagem resumida de reparações por técnico

Fluxo Normal

1. Gestor pede listagem de reparações por técnico para um dado mês
2. Sistema obtém lista de reparações do mês
3. Sistema calcula duração média das reparações
4. Sistema calcula média de desvio em relação às durações previstas

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor pede listagem de reparações por técnico para um dado mês	UI			
Sistema obtém lista de reparações do mês		obter todas reparações de um determinado mês, agrupadas por técnico	getReparacoesMes(data: LocalDate, tempo: Map<Tecnico, ReparaçãoPorMes>)	SubReparacoes
Sistema calcula duração média das reparações		obter duração média das reparações	obterDuracaoReparacoes() : Double	SubReparacoes
Sistema calcula média de desvio em relação às durações previstas		obter media de desvio	obterMediaDesvio() : Double	SubReparacoes

Tabela 3.18: Análise Use Case - Listar reparações resumida por técnico (ver 3.2.19)

3.2.20

Listagem exaustiva de todas as intervenções realizadas por um técnico.

Use Case Listar reparações detalhada por técnico

Cenários 5

Pré-condição Gestor autenticado e reparações efetuadas

Pós-condição Listagem detalhada de reparações por técnico

Fluxo Normal

1. Gestor pede listagem de reparações por técnico para um dado mês
2. Sistema obtém lista de reparações do mês
3. Sistema comunica todas as intervenções realizadas

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor pede listagem de reparações por técnico para um dado mês	UI			
Sistema obtém lista de reparações do mês		obter lista de reparação de um determinado mês, agrupado por técnico	getReparacoesMes(data: LocalDateTime) : Map<Tecnico, ReparaçãoPorMes>	SubReparacoes
Sistema comunica todas as intervenções realizadas	UI			

Tabela 3.19: Análise Use Case - Listar reparações detalhada por técnico (ver 3.2.20)

3.2.21

Extração da lista de equipamentos recebidos e entregues por funcionário de balcão, ordenadas por funcionário de balcão e filtrado por data.

O gestor indica as datas entre as quais quer que a avaliação do funcionários seja feita, pelo que anteriormente não estava definido. Adicionalmente, houve alterações nos métodos em que retornam os funcionários.

Use Case Listar fluxo de trabalho funcionário de balcão

Cenários 5

Pré-condição Gestor autenticado

Pós-condição Listagem de equipamentos recebidos e entregues pelo funcionário de balcão

Fluxo Normal

1. Gestor pede listagem de equipamentos recebidos por cada funcionário de balcão
2. Gestor indica as datas de início e fim da avaliação
3. Sistema obtém lista de equipamento recebidos para cada funcionário
4. Sistema obtém lista de equipamento entregue para cada funcionário

Use Case	Fluxo	Responsabilidade	API	Subsistema
Gestor pede listagem de equipamentos recebidos por cada funcionário de balcão	UI			
Gestor indica as datas de início e fim da avaliação	UI			SSColaboradores
Gestor pede listagem de equipamentos entregue por cada funcionário de balcão	UI			
Sistema obtém lista de equipamento recebidos para cada funcionário		Obtém lista de equipamentos entregues, agrupado por funcionário	getEquipRecebidos(LocalDate Time de, LocalDate Time ate): Map<FuncionárioBalcão, List<Equipamento>	SSColaboradores
Sistema obtém lista de equipamento entregue para cada funcionário		Obtém lista de equipamentos entregues, agrupado por funcionário	getEquipEntregue(LocalDate Time de, LocalDate Time ate): Map<FuncionárioBalcão, List<Equipamento>	SSColaboradores

Tabela 3.20: Análise Use Case - Listar fluxo de trabalho funcionário de balcão (ver 3.2.21)

4 Modelação Conceptual

4.1 Diagrama de Componentes

A partir da análise dos requisitos funcionais, em particular dos casos de uso, decidiu-se construir um Diagrama de Componentes (4.1) que visa, neste caso, estruturar o projeto por camadas, correspondendo estas a sistemas:

ReparacoesUI Camada de interação com o utilizador.

ReparacoesLN Camada de lógica do negócio, engloba todo o conhecimento sobre estado da aplicação, assim como o manusear.

ReparacoesBD Camada de acesso a base de dados.

Neste projeto, esta camada foi simplificada devido à falta de conhecimento dos elementos

ReparacoesUI

Como já referido, refere-se à camada de interação com o utilizador. Utiliza API da **ReparacoesLN** (ver 4.1).

Desta forma, o modo de interação com o utilizador pode ser modificado na sua totalidade sem que haja necessidade de qualquer alteração ao nível da camada de lógica.

ReparacoesLN

O sistema da lógica do negócio, expõe uma interface designada *IReparacoesLN*, por forma a garantir o encapsulamento do mesmo. Esta estratégia permite que o sistema que o utiliza (neste caso, **ReparacoesUI**) não dependa da sua implementação, isolando a implementação lógica. Não obstante, esta divisão inicial, foi identificado um conjunto de subsistemas que permitiram uma melhor organização, implementação e manutenção do projeto, procu-

rando diminuir as suas dependências. Assim, identificou-se três subsistemas: SSClientes, SSReparacoes, SSColaboradores. Utiliza a camada de base de dados (ver 4.1) por forma a garantir a persistência do estado da aplicação entre utilizações.

SSClientes Realiza a interface *IGestClientes*.

Engloba a lógica relacionada com os Clientes (2.2.1) e os seus Equipamentos (2.2.3).

SSColaboradores Realiza a interface *IGestColaboradores*.

Engloba toda a lógica relacionada com os Colaboradores (2.2.2) e as suas especificações e o balcão. Tem, ainda, a Gestão da Agenda dos Colaboradores. Esta funcionalidade derivou da necessidade dos técnicos terem uma lista de reparações e passar a ser possível fazer uma estimativa do prazo de reparação, sempre que é adicionada uma nova.

SSReparacoes Realiza a interface *IGestReparacoes*.

Engloba toda a lógica relacionada com as Reparações (2.2.5) e com os Orcamentos (2.2.4).

ReparacoesBD

Conforme referido, a camada de base de dados deste projeto corresponde, na realidade, a uma simples persistência do estado de toda a aplicação armazenado num ficheiro de objetos.

No entanto, optou-se por colocar uma camada específica para este efeito por forma a manter a garantia de facilidade de manutenção e uma eventual implementação de um sistema de base de dados.

Para tal, realiza a interface **IReparacoesBD**.

4.2 Diagrama de Classes

Utilizando a estrutura apresentada no Diagrama de Componentes (ver 4.1), desenvolveu-se vários Diagramas de Classe para representar as classes que compõe cada um dos sistemas. Atendendo que o sistema com maior complexidade é o *ReparacoesLN*, foi neste que o grupo se debruçou mais afincadamente. Utilizou-se uma estratégia **Facade** a estrutura desta classe.

Como já referido, através da análise dos casos de uso, identificou-se um conjunto de métodos necessários para a sua implementação. Estes métodos foram colocados ao nível da API *IReparacoesLN*. Posteriormente, implementou-se a interface através da classe *ReparacoesLN-*

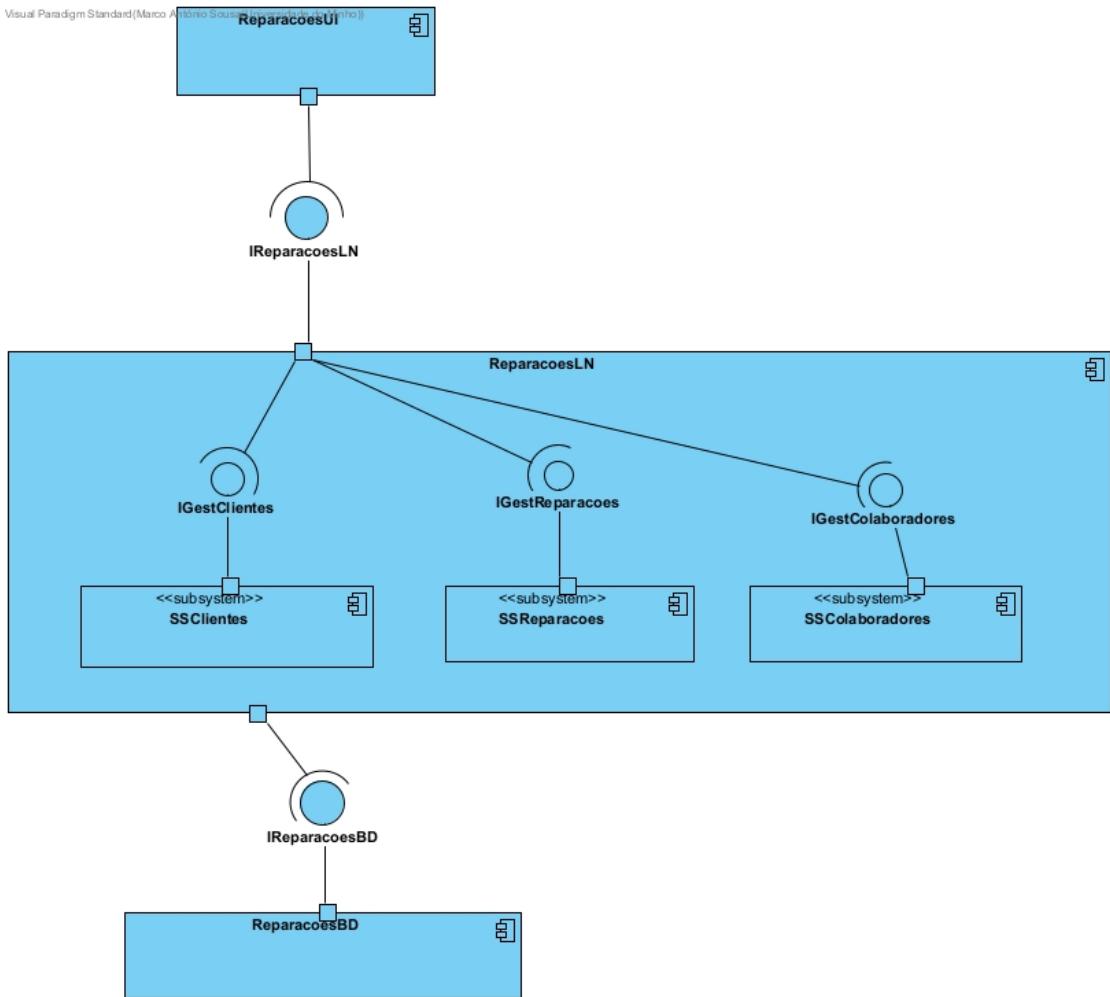


Figura 4.1: Diagrama de Componentes

Facade. Sendo este o ponto de entrada na camada de negócio, deve conhecer cada um dos subsistemas necessários. Assim, criou-se uma variável de instância para cada uma das API dos subsistemas: *IGestClientes* (ver 4.2), *IGestReparacoes* (ver 4.2) e *IGestColaboradores* (ver 4.2). Desta forma, passa a ser possível utilizar a estratégia de **Facade** para realizar a implementação dos casos de uso propostos, i.e. tem-se uma classe agregadora que permite um ponto de entrada na camada da lógica.

Para a construção das classes necessárias para suportar as funcionalidades requisitadas, começou por se analisar o Modelo de Domínio (2) e converter as entidades em classes. Após uma fase inicial, para além de se ter verificado que a modelação de domínio realizada na fase I não era suficiente, foi, ainda, necessário criar algumas **classes de implementação** para

melhorar a eficiência e facilitar a implementação, tais como *Precos*, *CustoTotalReparacao*, *ReparacoesPorMes* e *GestAgenda*. Estas classes serão abordadas nos respetivos sistemas.

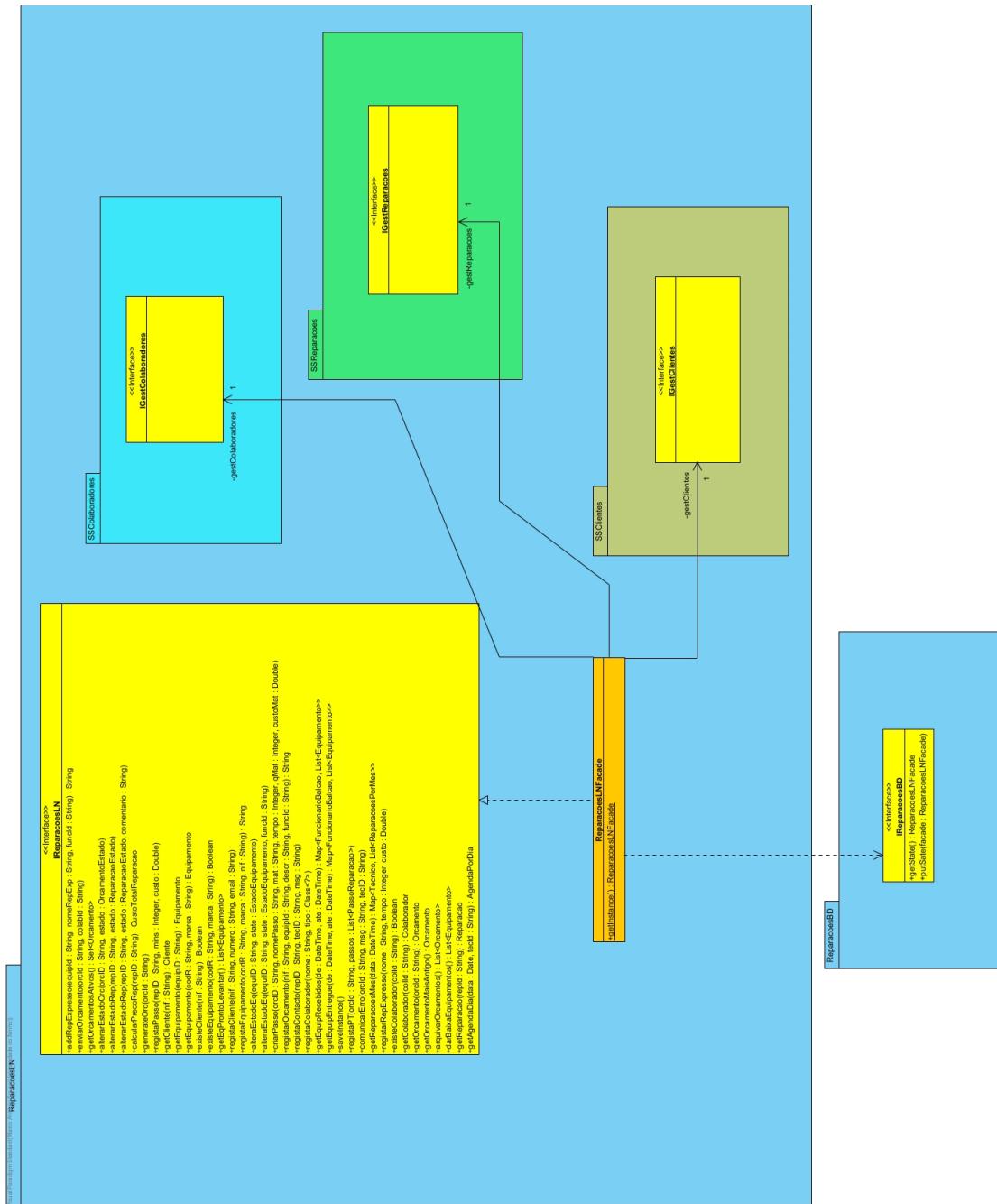


Figura 4.2: Diagrama de Classe - ReparacoesLN

SS Clientes

Neste subsistema podem-se encontrar as funcionalidades relacionadas com os **Clientes** do sistema. Apesar de cada subsistema não estar intrinsecamente dependente de outros, estes não deixam de se relacionarem, como se pode verificar neste caso, com o **SSReparacoes**.

Este subsistema define todas as dinâmicas relativas aos clientes do Centro de Reparações e as suas interações, ao nível do sistema desenvolvido, com as classes que engloba e as classes de outros subsistemas da arquitetura.

Encontram-se as classes principais:

- Cliente
- Equipamento

Cliente

A classe cliente representa, como referido anteriormente, os clientes do centro de reparações. Estes possuem um ou mais equipamentos associados a sí, sendo que possuí um **Map** para os guardar, sendo a chave o seu código de registo. Adicionalmente têm associados a sí uma classe de **FormaContacto** para guardar os seus contactos, *Email* e *número*.

Equipamento

De um modo bidirecional, os equipamentos tem acesso aos seus proprietários, para facilitar a obtenção destes. Adicionalmente ao seu identificador, código de registo e marca, o equipamento encontra-se associado ao um enum - **EstadoEquipamento**, para parametrizar os vários estados do equipamento no decorrer da sua estadia no centro de reparações. Como cada equipamento tem um histórico de reparações associado a ele, houve a necessidade de o equipamento estar relacionado com o subsistema **SSReparacoes** (Ver 4.2), na classe **Reparacao**. Por necessidade esta relação é bidirecional, para permitir à reparação ter conhecimento do equipamento que está a reparar.

GestClientesFacade

Esta classe permite a gestão dos clientes e dos respetivos equipamentos associados, implementando a interface **IGestClientes**. Esta classe possui todos os clientes e equipamentos do

sistema.

Analizando os métodos que esta classe implementa, identificou-se que, para melhorar a naveabilidade e a *performance* do sistema, será necessário que este tenha um conjunto de atributos, nomeadamente:

clientes Mapa de clientes que pode ser acessido através do seu identificador, nif

equipamentos Mapa de equipamentos que pode ser acessido através do seu ID

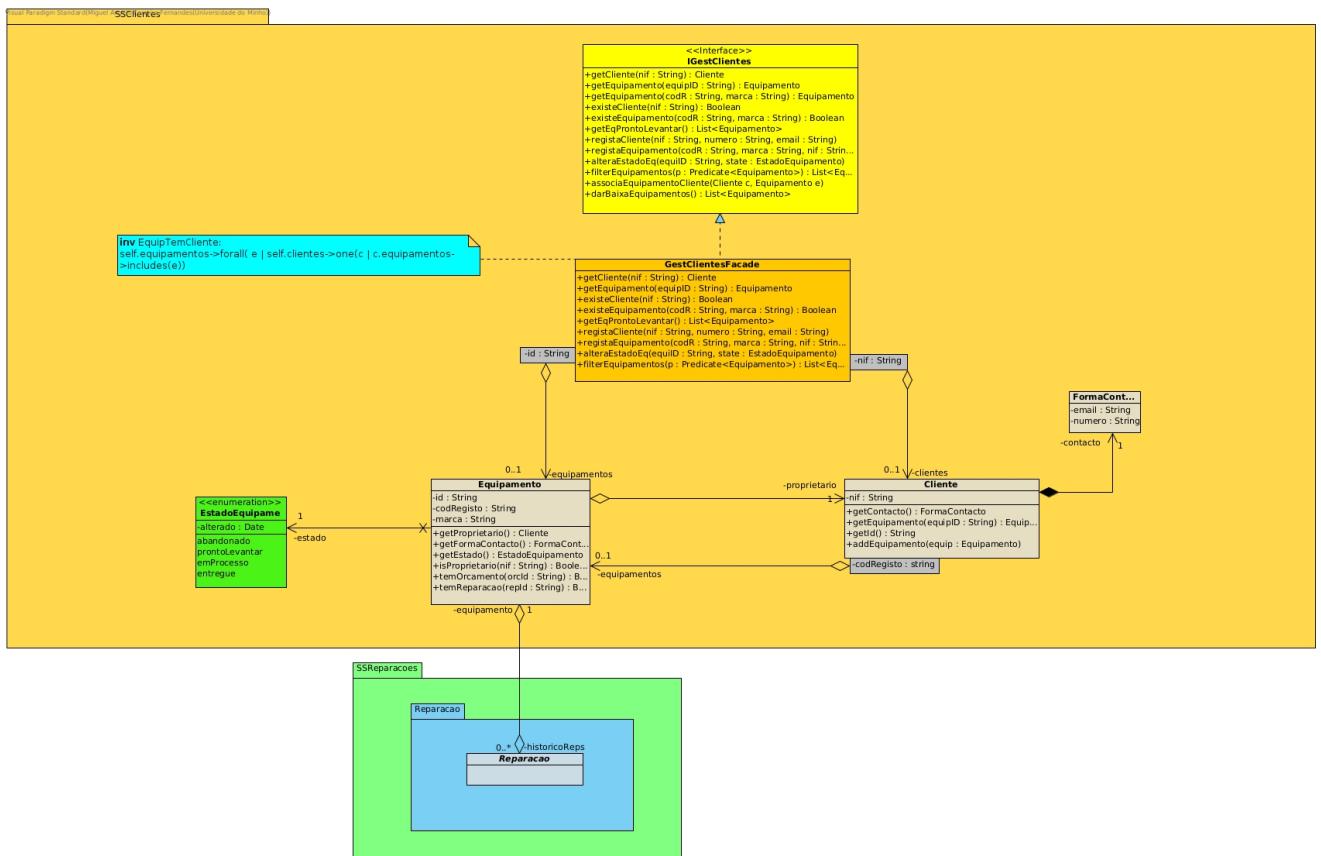


Figura 4.3: SubSistema SSClientes 4.2

SS Colaboradores

Neste subsistema podem encontrar-se as funcionalidades relacionadas com os **Colaboradores**. Existem 3 tipos de colaboradores, nomeadamente:

- Funcionário do Balcão

- Colaborador Especializado (Gestor e Técnico)

Neste subsistema, encontra-se apresentada toda a dinâmica e interações entre os elementos intervenientes do centro de reparações, os colaboradores, e os outros subsistemas da arquitetura.

Para facilitar todo o trabalho por parte do funcionário do balcão foi criado o *package* Balcão, que é responsável pela gestão da receção e entrega dos equipamentos guardando as respetivas datas, sendo visível a partir deste a relação entre os colaboradores e o equipamento, o subsistema **SSClientes**.

Por outro lado, para facilitar toda a gestão de tempo e/ou disponibilidade por parte dos colaboradores especializados, nomeadamente, o Técnico, na realização das diversas reparações, foi criado um *package* Agenda.

GestColaboradoresFacade

Esta classe é responsável por toda a gestão dos colaboradores e funcionalidades, implementando a interface **IGestColaboradores**.

Analizando os métodos que esta classe implementa, identificou-se que, para melhorar a naveabilidade e a *performance* do sistema, será necessário que este tenha um conjunto de atributos, nomeadamente:

`balcao` Map de interações entre o Funcionário de Balcão e o cliente

`colabs` Map de colaboradores que pode ser acessido através do seu ID

`agenda` Map para guardar todas as agendas dos respetivos técnicos

SS Reparações

Neste subsistema podem encontrar-se funcionalidades relacionadas com as **Reparações**. Este, por sua vez, está dividido em 3 *packages*:

`Orçamento` Responsável por atribuir um orçamento a um equipamento. Aqui, verifica-se que o orçamento tem um estado, facilitando o processo e dando um ponto de situação sobre o mesmo. Está também associado ao *package* plano de trabalho, sendo gerado a partir deste, e a uma comunicação, que tem como função interagir com o cliente se necessário. É de salientar a relação com um outro subsistema, **SSClientes**, devido ao

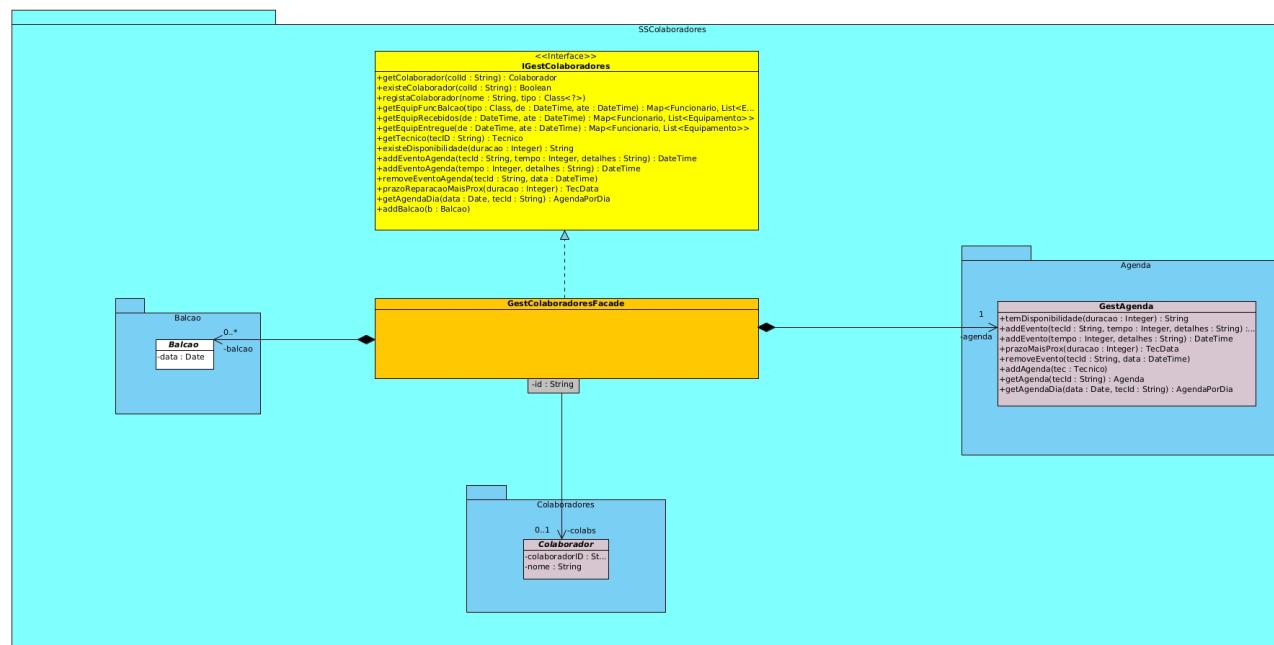


Figura 4.4: SubSistema SSReparacoes 4.2

atributo do equipamento.

Reparação Responsável pela reparação do equipamento. Existem dois tipos de reparação: Reparação Programada e a Reparação Expresso. Cada reparação tem um estado, de forma a facilitar o processo e dando um ponto de situação, e está associado a um orçamento, a um plano de trabalhos e a uma lista de comunicações que serão feitas, caso sejam feitas, com o cliente. É de salientar a existência da classe ReparacoesPorMes, na medida que esta foi criada a facilitar as listagens dos colaboradores.

Plano de Trabalho Responsável pelo planeamento dos passos e subpassos de uma reparação. Aqui, verifica-se que o plano de trabalhos tem como atributos duas listas distintas de passos de reparação. Por sua vez, esta classe, passo de reparação, está associada ao material. A classe material representa todos os materiais usados pelo passo de reparação, ou seja, é impossível categorizar um conjunto variado de materiais, porém, a classe CategoriaMaterial encontra-se presente na mesma, sendo que, é uma possível melhoria neste trabalho, isto é, a classe Material passaria a representar não um conjunto de materiais, mas apenas um, ao qual estaria associado uma categoria. Daqui também se poderia aprofundar a ideia do stock da loja.

4.2.1 GestReparaçõesFacade

Esta classe é responsável por toda a gestão das reparações e funcionalidades, implementando a interface `IGestReparacoes`.

Analisando os métodos que esta classe implementa, identificou-se que, para melhorar a naveabilidade e a *performance* do sistema, será necessário que este tenha um conjunto de atributos, nomeadamente:

`reps` Map de Reparações por realizar

`reparacoesDisponiveis` Set de Reparacoes Expresso existentes no sistemas

`orcs` Map de Orçamentos que podem ser acessidos através do seu ID

Classes de Implementação

Foram construídas várias classes de implementação, nomeadamente: `Precos`, `CustoTotalReparacao`, `ReparacoesPorMes` e `GestAgenda`.

Precos

Constituída unicamente por variáveis e métodos estáticos, permite armazenar o custo da hora do técnico. O objetivo desta classe é que ao criar um novo orçamento, seja copiado para o orçamento o valor horário do técnico e garantir que, mesmo havendo flutuações do custo da mão de obra, estes não irão afetar o valor do orçamento.

Optou-se por colocar desta forma para que haja maior facilidade de acesso ao valor, por qualquer classe.

CustoTotalReparacao

Classe auxiliar que permite calcular o preço total de uma reparação, assim como o desvio de tempo em relação ao estimado. A sua utilização consiste em, depois de ser instanciada, adicionar um novo passo (`tempo`, `tempoEstimado` e `custoMaterial`).

Sempre que recebe um novo passo, os parâmetros recebidos são adicionados aos acumuladores de instância.

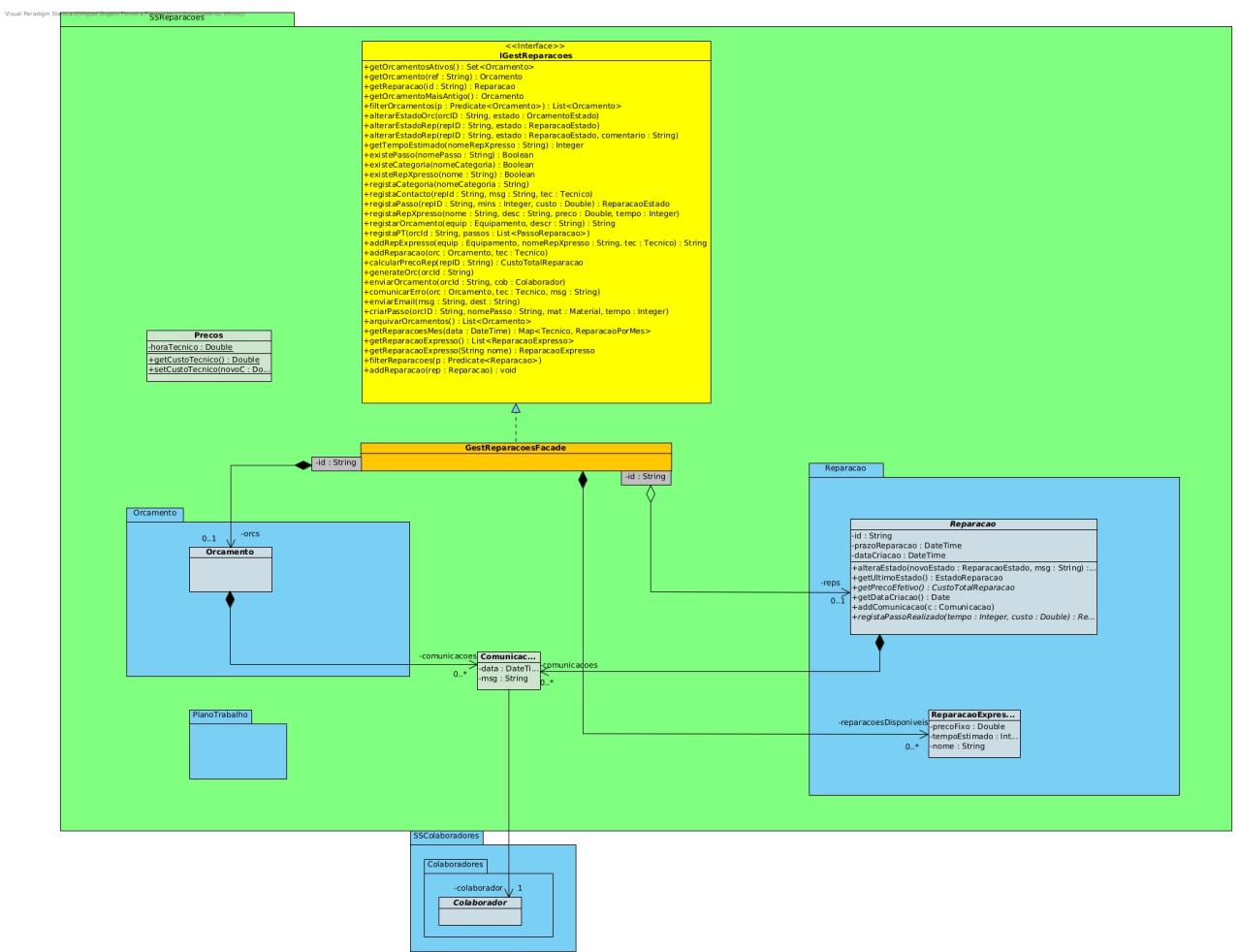


Figura 4.5: SubSistema SSReparacoes 4.2



Figura 4.6: Precos

No fundo, esta classe serve como um acumulador que centraliza as operações e devolve várias informações sobre a reparação.

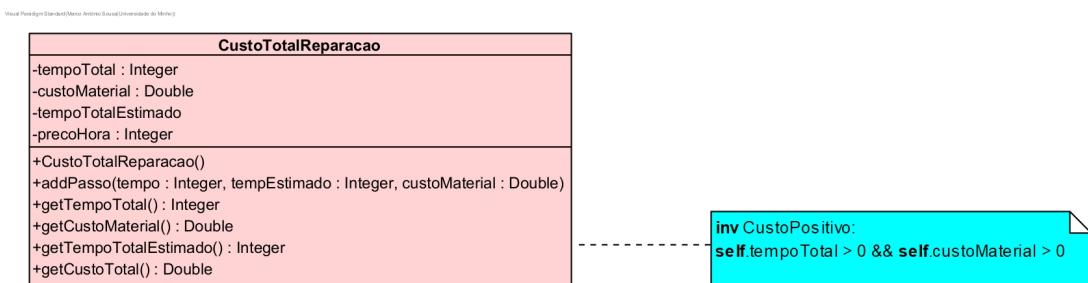


Figura 4.7: Custo Total Reparacao

ReparacoesPorMes

Tal como o anterior, esta classe também tem como finalidade servidor de acumulador. No entanto, trata-se de um acumulador um pouco mais complexo, no sentido em que sempre que é adicionado uma nova reparação, calcula a sua duração e atualiza duas médias: duração média das reparações e média do desvio da duração das reparação em relação ao esperado. Para tal, é utilizado a média ponderada entre o acumulador e novo valor a acrescentar, mantendo uma variável auxiliar que é o total.

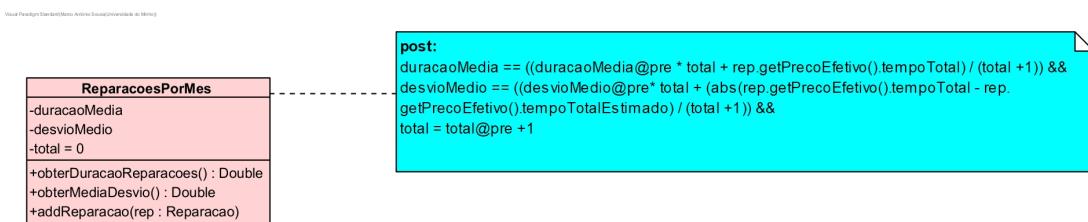


Figura 4.8: Reparacoes Por Mes

GestAgenda

Por forma a organizar o trabalho dos técnicos, desenvolveu-se uma classe para esse efeito. Para a sua implementação, foi necessário criar 5 classes.

GestAgenda Efetua a gestão de todas as agendas, havendo uma agenda por cada técnico.

Serve como uma API para a gestão das agendas.

Por forma a garantir a distribuição de carga de trabalho entre os vários técnicos, utilizou-se uma *Queue* de agendas. Assim, sempre que se adiciona um novo evento, a agenda do técnico passa para último. Aumentando a probabilidade que o próximo

técnico ao qual se vai tentar atribuir trabalho seja outro.

Para obter o prazo mais próximo, é iterado cada agenda e pede-se à mesma o prazo mais próximo. Desta forma, retorna-se aquele que tiver o prazo estimado mais baixo.

Agenda Atribuída uma a cada técnico e armazena os eventos organizados por dia.

AgendaPorDia Armazena todos os eventos para um dado dia. É utilizado pela Agenda.

EntradaAgenda Como o nome indica, corresponde a uma entrada na agenda do técnico.

TecData Classe auxiliar que serve para transportar o *tuple* {tecid: String, data: LocalDate-Time}.

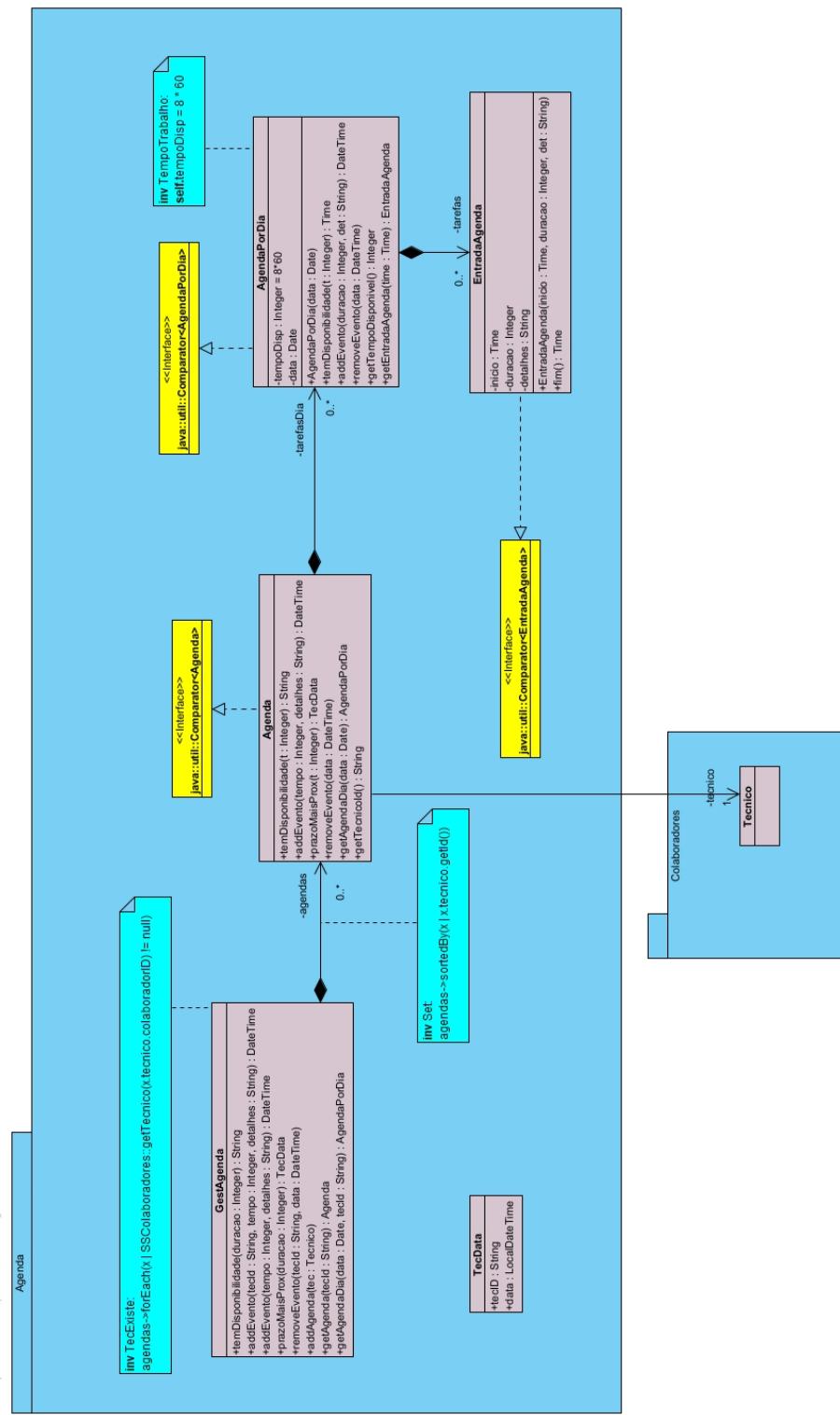


Figura 4.9: Agenda

4.3 Diagramas de Sequência

De forma a ilustrar alguns dos diagramas de sequência desenvolvidos, foram escolhidos 3 use cases com o objetivo de apresentar todos os diagramas referentes a estes e também qual foi a estratégia escolhida para os resolver. Foram considerados 3 *uses cases* "principais", nomeadamente o Fazer Orçamento, o Realizar Reparação e ainda o Pedir Reparação Expresso.

4.3.1 Fazer Orçamento

Tal como é referido no use case, quando o técnico pretende fazer um orçamento, este solicita o pedido de orçamento mais antigo. O sistema para responder a este pedido, constrói um Set ordenado por ordem descrescente da data de criação com todos os Orçamentos em que o estado é "porCalcular". Set este construído a partir do método `getOrcamentosAtivos()`, chamado pelo `getOrcamentoMaisAntigo()` que lhe retira o primeiro elemento.

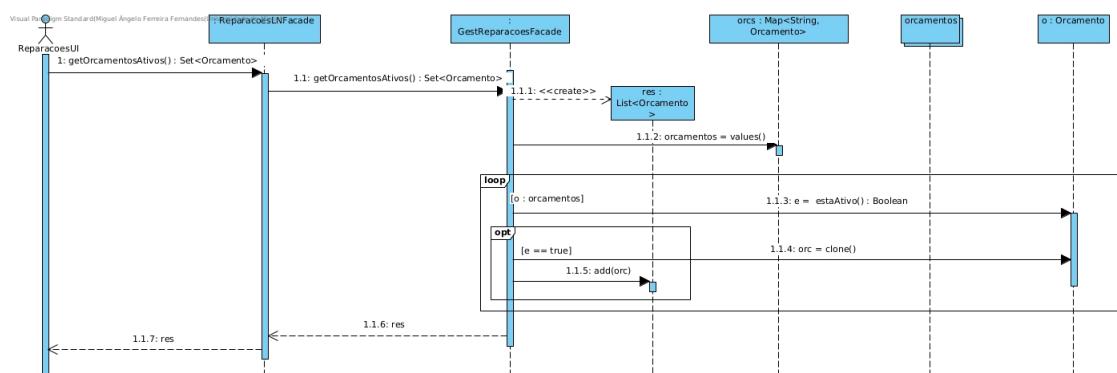


Figura 4.10: Diagrama de sequência `getOrcamentosAtivos`

Depois de saber qual o Orçamento que tem de calcular, o técnico analisa o equipamento e vai construindo o plano de trabalhos adicionando sequencialmente os passos necessários para a sua reparação. Cada passo é identificado por um nome e é caracterizado pelo tempo estimado e ainda por o material que é expectável usar.

Quando todos os passos já foram adicionados e o plano de trabalhos já foi concluído o técnico pode gerar o Orçamento e registar o envio do mesmo ao cliente. O estado do orçamento muda de "porCalcular" para enviado e é adicionado às comunicações do Orçamento o contacto referente ao envio.

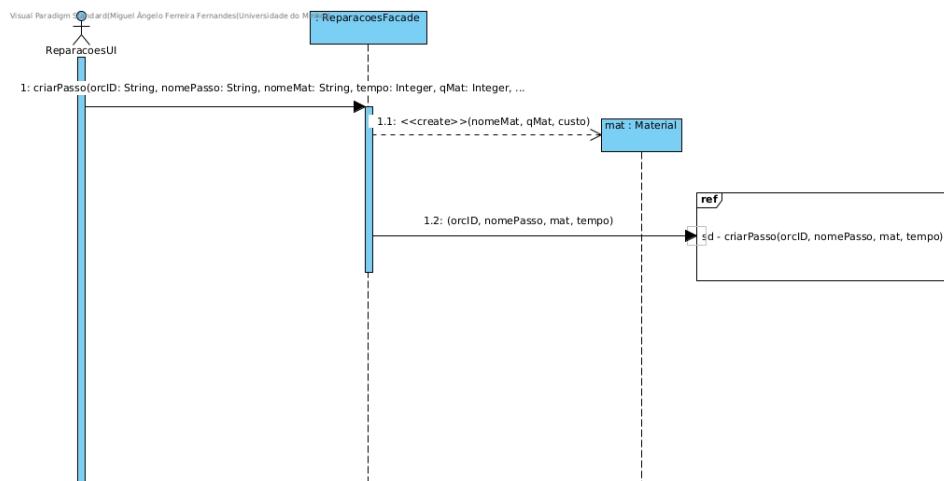


Figura 4.11: Diagrama de sequência criarPasso(1)

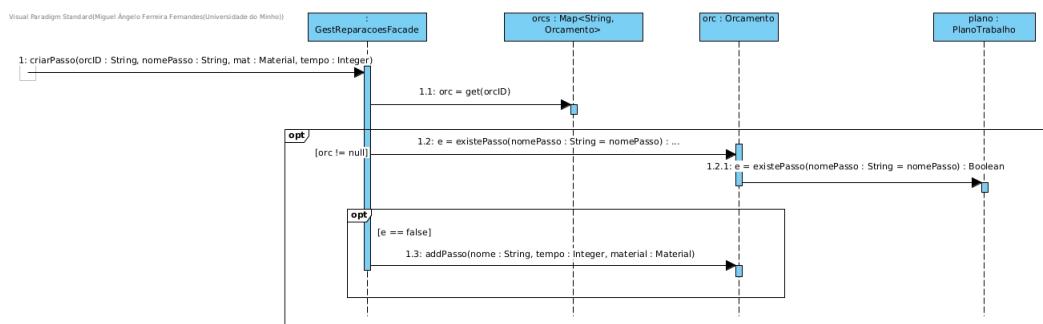


Figura 4.12: Diagrama de sequência criarPasso(2)

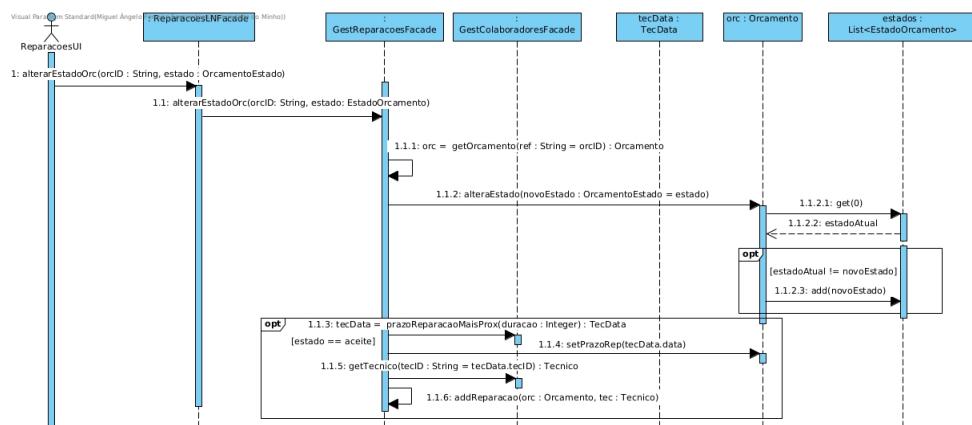


Figura 4.13: Diagrama de sequência alterarEstadoOrc

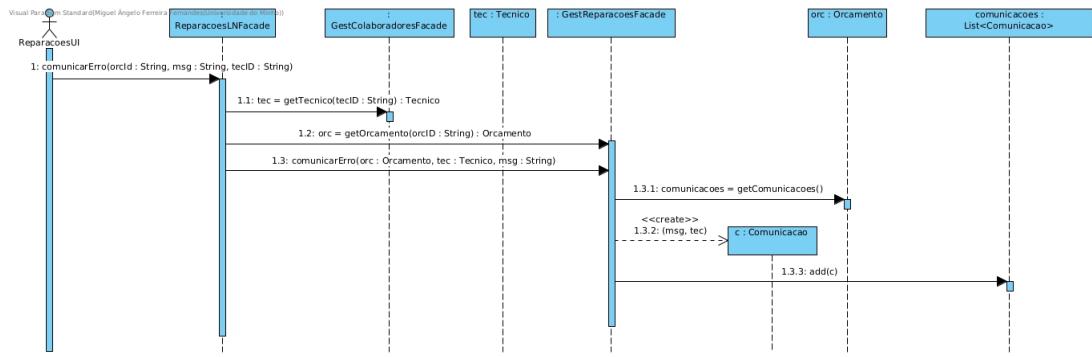


Figura 4.14: Diagrama de sequência comunicarErro

4.3.2 Realizar Reparação

Uma reparação quando é criada é desde logo atribuída a um técnico e é registado na sua agenda a hora a que este deve a começar e ainda a hora que tem de terminar. Tendo em conta isto, todos os técnicos têm uma agenda e é nesta que vão buscar os identificadores das reparações que lhe foram atribuídas. Logo que o técnico inicie o processo de reparação o estado desta tem de ser alterado para "emReparacao".

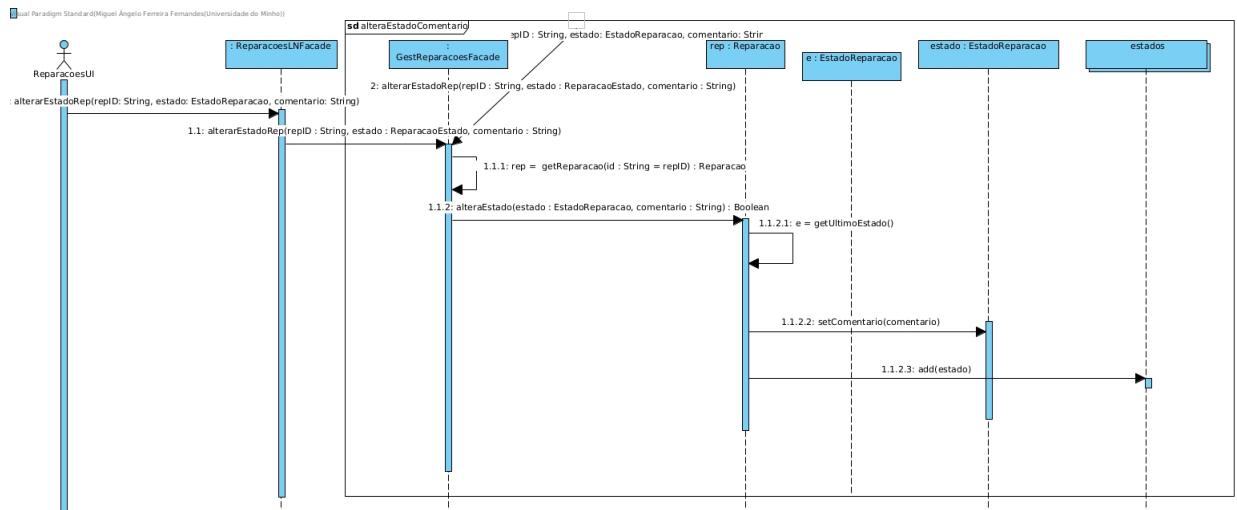


Figura 4.15: Diagrama de sequência AlterarEstadoRep

Após o estado mudar, caso se trate de um reparação programada o técnico vai realizando, sequencialmente, os passos da reparação do plano de trabalhos, acrescentando estes aos passos realizados e indicando o tempo gasto e ainda o custo efetivo. Quando não existe mais passos para serem realizados a reparação é dada como concluída e o equipamento

passa a estar pronto para levantar. No outro caso, em que a reparação é expresso, o técnico limita-se a registar a conclusão alterando, da mesma forma que a anterior, o estado da reparação e o estado do equipamento.

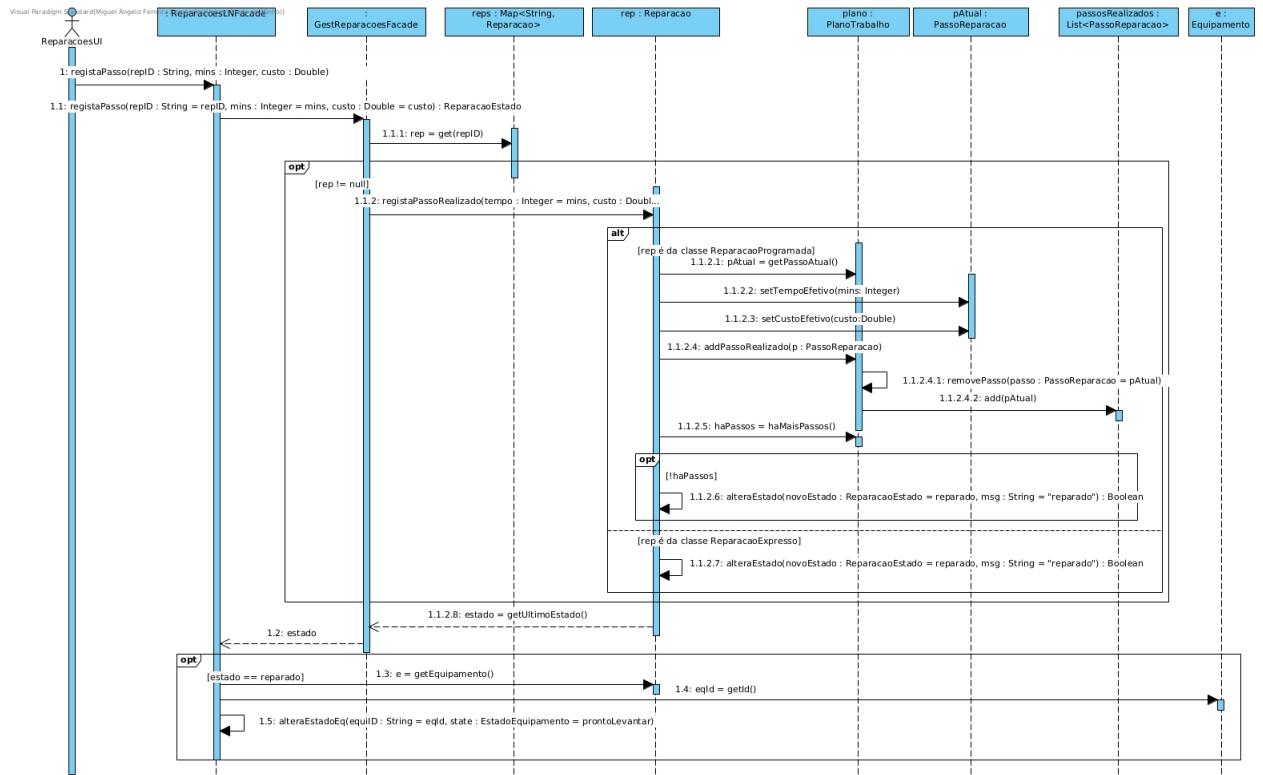


Figura 4.16: Diagrama de sequência registrarPasso

4.3.3 Pedir Reparação Expresso

O funcionário de balcão é o responsável por realizar os pedidos de reparação expresso. Após identificar o cliente e o equipamento deste, é inserido no sistema o nome da reparação expresso a ser realizada. Após a insersão de dados, em primeiro lugar é verificado se o nome corresponde a alguma reparação expresso disponível no sistema. Caso esta verificação seja sucedida, o sistema verifica se existe disponibilidade para realizar a reparação de imediato, isto é, até à hora do fim do centro de reparações. Para o sistema concluir se existe disponibilidade, este percorre as agendas de todos os técnicos e verifica se existe algum "buraco", onde seja possível encaixar a reparação pedida. Caso se verifique disponibilidade, é retornado qual o técnico atribuído e a reparação é aceite, caso contrário a reparação é recusada. De maneira a que receções de equipamentos por partes dos funcionários de balcão fiquem registadas, é

ainda no método addRepExpresso passado o identificador do funcionário responsável para que seja criada uma instância da classe balcão que será guardada numa lista que mais tarde pode ser analisada pelo gestor.

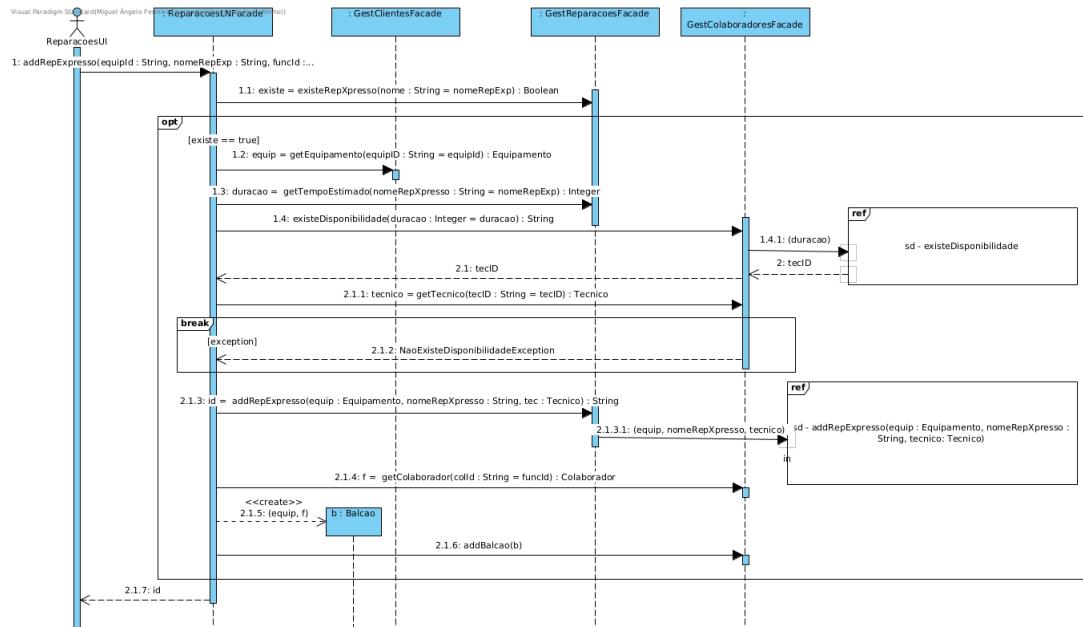


Figura 4.17: Diagrama de sequência addRepExpresso(1)

4.4 Diagrama de Package

Durante a implementação, começou a tornar-se confuso gerir o número crescente de classes. Por esse motivo, decidiu organizar-se em *Packages*.

A primeira versão da organização, consistiu, essencialmente, em "transformar" o diagrama de componentes em pacotes. Posteriormente, identificou-se um conjunto de contextos que permitiram uma melhor organização das classes.

SSClientes Corresponde ao sistema de gestão de clientes.

Tratando-se de um sistema pequeno, não se efetuou nenhuma subdivisão.

SSReparacoes Identificou-se três contextos de utilização:

Reparação Toda a lógica relacionada com a Reparação, ReparaçãoProgramada e ReparaçãoExpresso.

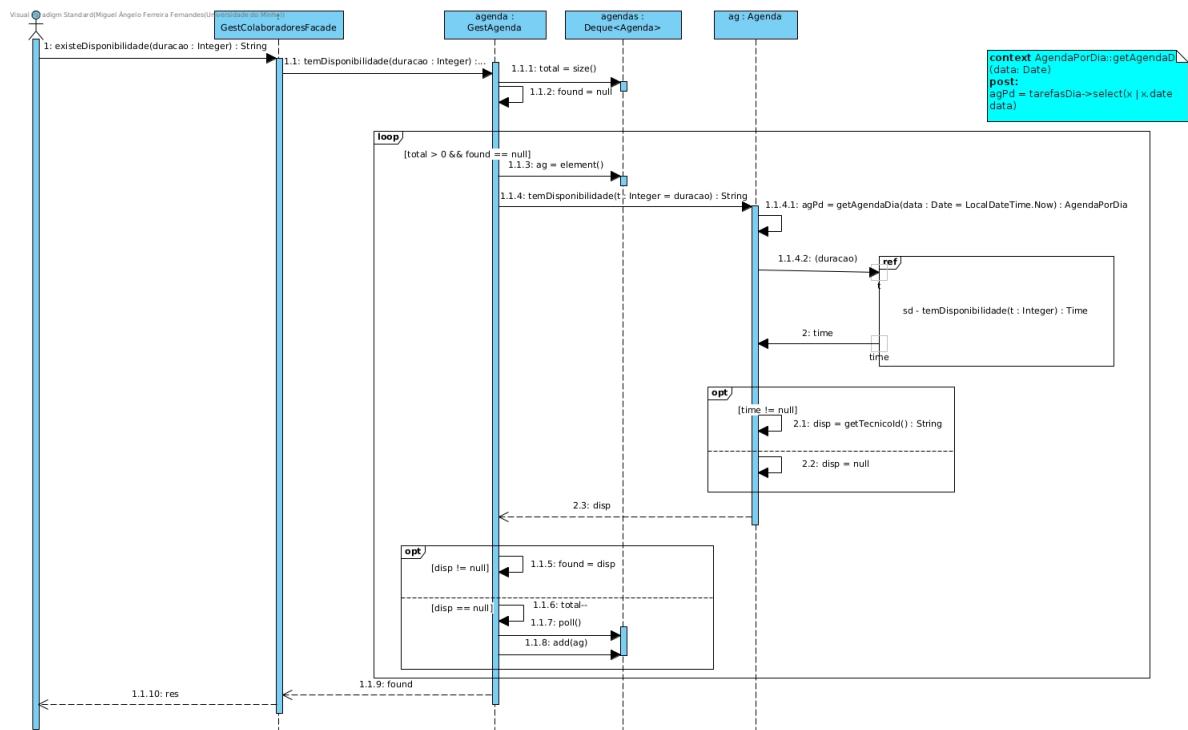


Figura 4.18: Diagrama de sequência existeDisponibilidade

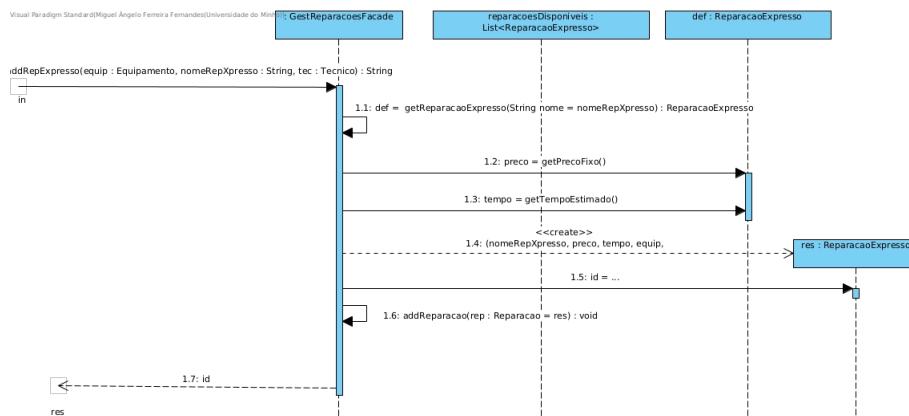


Figura 4.19: Diagrama de sequência addRepExpresso(2)

Colocou-se, ainda, as classes de implementação utilizadas por si (CustoTotalReparacao e ReparacoesPorMes).

Orcamento Tal como o nome indica, corresponde às classes que implementam a lógica do Orçamento.

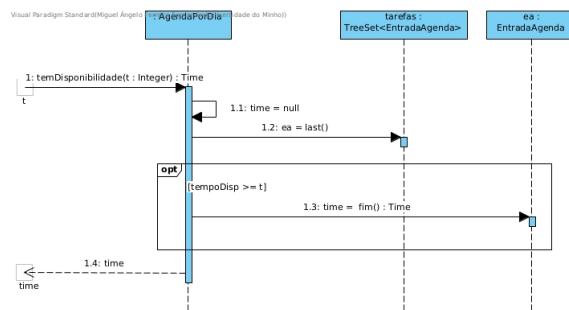


Figura 4.20: Diagrama de sequência temDisponibilidade

PlanoTrabalho Sendo este um pacote comum aos dois anteriores, optou-se por colocar como um próprio, passando os outros a utilizá-lo.

SSColaboradores Identificou-se, também, três contextos:

Colaboradores Gestão dos colaboradores (adicionar ou alterar).

Balcao Corresponde à interação do Funcionário de Balcão com os Equipamentos (Receber ou Entregar). Optou-se por colocar num pacote próprio para tornar as dependências menos "aninhadas".

Eventualmente, poderia colocar-se esta lógica junto dos Colaboradores.

Agenda Atendendo que tem uma implementação isolada, é fácil de perceber que se encontra num contexto próprio e, portanto, foi-lhe atribuída um pacote.

Ao utilizar a estrutura definida, ficou mais percutível o papel de cada classe. Dizendo apenas o sistema em que se encontra, torna-se ineficiente e menos compreensivo, pois pode não se conseguir atribuir, de imediato, uma classe a um contexto.

Será de referir que o diagrama de pacotes (4.21) desenvolvido se encontra incompleto, por dúvida sobre a sua construção.

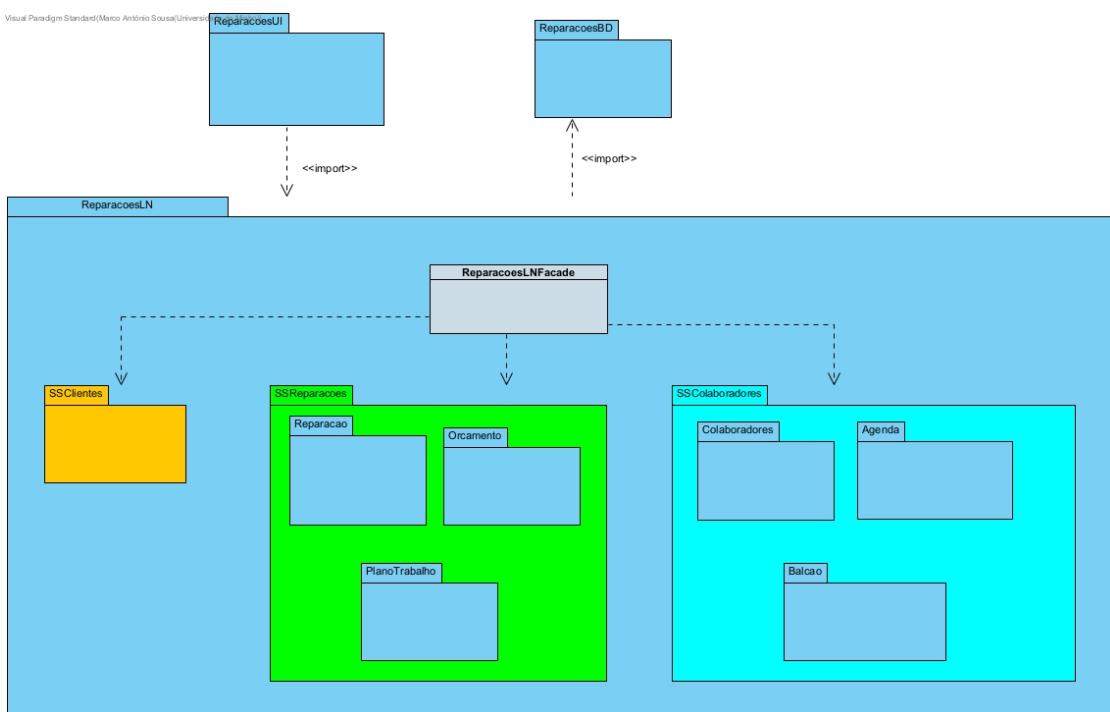


Figura 4.21: Diagrama de Package

5 Implementação

Para o desenvolvimento deste projeto, foi utilizada uma estratégia de programação orientada aos modelos, i.e. foi desenvolvido a maioria dos diagramas de sequência e de classe, posteriormente procedeu-se à implementação do código.

Esta estratégia permitiu que se antecipasse bastantes problemas de métodos (assinaturas, valores retornados), assim como identificar classes de implementação necessárias. Apesar disso, ao longo da implementação denotou-se ainda algumas falhas que obrigaram a alterações pontuais.

No entanto, fruto da estratégia utilizada a implementação do código foi realizada em aproximadamente 2/ 3 dias, conforme distribuído através de um diagrama de Gantt (ver 5.1).

Para facilitar o trabalho em grupo, utilizou-se um repositório no *GitHub* para efetuar *source control* e o programa *Visual Paradigm* com a funcionalidade de *Team* e *Tasifier*. Este último, revelou-se fundamental, pois permitiu a distribuição das tarefas pelos elementos e o feedback sobre o mesmo.

Ao nível de código, utilizou-se um projeto *Maven* para gerir os *imports* necessários, principalmente ao nível de testes. Infelizmente, não foi possível desenvolver testes suficientes devido à falta de tempo, ficando pendente os testes relacionados com o GestReparacoesFacade.

ID	Title	Start Time	End Time	Sun 12-26	Mon 12-27	Tue 12-28	Wed 12-29	Thu 12-30	Fri 12-31	Sat 01-01
47	Orcamento	2021-12-27	2021-12-29		Orcamento					
48	PlanoTrabalho	2021-12-27	2021-12-28		PlanoTrabalho					
56	Cliente	2021-12-27	2021-12-28		Cliente					
51	Reparacao	2021-12-27	2021-12-28		Reparacao					
53	ReparacaoEx...	2021-12-27	2021-12-28		ReparacaoExpresso					
58	FormaContacto	2021-12-27	2021-12-28		FormaContacto					
59	▲ Colaborador	2021-12-28	2021-12-30							
60	colaborador...	2021-12-28	2021-12-30		colaborador especializado					
61	technico	2021-12-28	2021-12-30		technico					
62	gestor	2021-12-28	2021-12-30		gestor					
63	Funcionario...	2021-12-28	2021-12-30		FuncionarioBalcão					
68	registrarRepEx...	2021-12-27	2021-12-28		registrarRepExpresso					
6	getOrcamento...	2021-12-26	2021-12-29		getOrcamentosAtivos					
7	alterarEstadoOrc	2021-12-26	2021-12-29		alterarEstadoOrc					
8	alterarEstado...	2021-12-26	2021-12-29		alterarEstadoRep					
9	alterarEstado...	2021-12-26	2021-12-29		alterarEstadoRep					
12	registraPasso	2021-12-26	2021-12-29		registraPasso					
17	getEqProntoL...	2021-12-26	2021-12-29		getEqProntoLevantar					
18	registraCliente	2021-12-26	2021-12-29		registraCliente					
20	alteraEstadoEq	2021-12-26	2021-12-29		alteraEstadoEq					
21	criarPasso	2021-12-26	2021-12-29		criarPasso					
22	registrarOrcam...	2021-12-26	2021-12-29		registrarOrcamento					

Figura 5.1: Recorte do diagrama de Gantt

5.1 Menus

Os menus foram desenhados de maneira a corresponder a todas as funcionalidades descritas nos *Use Cases*.

```
Bem vindo ao Sistema DSS-TechSupport!  
  
*** Menu ***  
1 - Autenticar  
0 - Sair  
Opção:
```

Figura 5.2: Menu inicial

```
*** Menu ***  
1 - Fazer orçamento  
2 - Realizar reparação  
3 - Menu Colaborador Especializado  
4 - Verificar agenda  
0 - Sair  
Opção:
```

Figura 5.3: Menu Técnico

```

Orcamento: 1
Descrição: Problema com 1
Equipamento: Identificador: 1
Marca: P00
Código Registo: 1

*** Menu ***
1 - Definir plano de trabalhos
2 - Gerar Orcamento
0 - Sair
Opção: 1
:::::::::::
Plano de Trabalhos atual: Passos do plano:

*** Menu ***
1 - Adicionar passo de reparação
2 - Registar comunicação com o cliente
0 - Sair
Opção: 2
:::::::::::
Opção: 2
:::::::::::
Orcamento mais antigo: Orcamento [comunicacoes =, custoHora =25.0, descrProb =Problema com 1, equipamento =Identificador: 1
Marca: P00
Código Registo: 1
, estado atual =EstadoOrcamento [data=2021-12-30T22:50:24.625414697, estado=porCalcular], id =1, plano =Passos do plano:
, prazoReparacao =null, preco =8.0, reparacao =null]

```

Figura 5.4: Menu Fazer orçamento

```

Insira nome do passo a adicionar:
trocar entrada usb
Insira o material a adicionar:
entrada usb
Insira quantidade de material:
1
Insira custo total do material:
3
Insira o tempo estimado:
5
Passo trocar entrada usb adicionado ao orçamento com o id: 2

```

Figura 5.5: Exemplo Adicionar Passo

```
*** Menu ***
1 - Registar Cliente
2 - Registar Equipamento
3 - Registar Pedido de Orçamento
4 - Confirmar Orçamento
5 - Registar Pedido de Reparação Expresso
6 - Registar Entrega do Equipamento
7 - Consultar Orçamentos enviados
0 - Sair
Opção:
```

Figura 5.6: Menu Funcionário Balcão

```
Insira o nif do cliente:
123456789
Insira o numero do cliente:
123456789
Insira o email do cliente:
teste@uminho.pt
Cliente registado com o nif:123456789
```

Figura 5.7: Exemplo registrar cliente

```
Insira o nif do cliente:
123456789
Insira a marca do equipamento:
Asus
Insira o código de registo do equipamento:
111222
Equipamento registado com o identificador: 100
```

Figura 5.8: Exemplo registrar equipamento

```
Insira o nif do cliente:
123456789
Insira o identificador do equipamento:
100
Insira a descrição do orçamento:
tela partida
Pedido de Orcamento registado com o id: 60
```

Figura 5.9: Exemplo pedir orçamento

```
Insira o identificador do Orcamento:  
1  
  
*** Menu ***  
1 - Confirmar Orcamento  
2 - Recusar Orcamento  
0 - Sair  
Opção: 1  
Orçamento 1 confirmado!
```

Figura 5.10: Exemplo confirmar orçamento

```
Insira o tipo de reparação expresso:  
Mudar bateria telemovel  
Insira o identificador do equipamento:  
100  
Não existe disponibilidade para realizar a reparação de imediato
```

Figura 5.11: Exemplo pedir reparação expresso

```
*** Menu ***  
1 - Registar Colaborador  
2 - Listar reparações detalhadas por técnico  
3 - Listar reparações resumidas por técnico  
4 - Listar fluxo de trabalho do funcionário de balcão  
5 - Menu Colaborador Especializado  
6 - Dar baixa de equipamentos  
7 - Arquivar orçamentos  
0 - Sair  
Opção:
```

Figura 5.12: Menu Gestor

6 Considerações Finais

6.1 Conclusões

Com o desenvolvimento do presente relatório, assim como as atividades inerentes ao mesmo, o grupo deparou-se com um conjunto de dificuldades relacionadas com a falta de experiência na área de Engenharia de Software. Como referido no relatório formulado na primeira fase, o grupo tinha consciência da sua inexperiência e teve de responder à necessidade de abstrair-se do que foi definido no relatório, de modo a encontrar uma solução adequada. Contudo, com a ajuda dos docentes e a partir da experiência obtida na realização de outros projetos da mesma índole, foi possível obter uma solução que satisfaz o problema proposto ao grupo. Devido ao seu foco durante o semestre letivo e a importância atribuída nas aulas lecionadas, foi colocado como prioridade o aperfeiçoamento da modelação final do projeto, havendo repercussões na implementação do projeto que, apesar de facilitada por este planeamento prévio, não encontra-se no estado final desejado.

A implementação responde, através de vários menus, às funcionalidades definidas nos vários casos de uso, seguindo a estrutura e comportamento definidos nos Diagramas de **Componentes, Classes e Sequência**. Contudo, existem várias inconsistências que o grupo não conseguiu eliminar até a entrega do trabalho. Estes problemas impedem a conceção completa da modelagem efetuada e não correspondem à solução imaginada.

Relativamente a modelação, procurou-se desenvolver as várias etapas, utilizando a maior formalidade possível. Para tal, aproveitou-se a linguagem UML que mune o grupo com as ferramentas necessárias para representar os conceitos com o nível de abstração suficiente, como recomendado pelos docentes da unidade curricular.

O grupo considera que, apesar das dificuldades encontradas no decorrer da sua realização, e, alguns precalços que seriam apagados em iterações futuras, respondeu ao problema posto.

A modelação

/sectionTrabalho Futuro

O projeto apresentado, e especificado no contexto deste relatório, apresenta uma solução para o problema proposto pelos docentes através dos cinco cenários apresentados. Apesar do grupo considerar que a solução desenvolvida responde às necessidades do Centro de Reparações, propõem, simultaneamente, alguns casos de estudo para iterações futuras da mesma, de modo a aumentar a sua completude e formular um sistema útil.

Um dos casos principais que seriam atacados é os referentes ao uso de materiais no sistema. O grupo planeou a manipulação de *stocks* num contexto real, sendo descartado na iteração presente do projeto. Com a implementação destas dinâmicas, com encomendar e adicionar *stock* de materiais à loja, bem como remover, proporcionaria um nível maior de gestão da loja.

No seguimento da implementação dos materiais no sistema, seria importante a adição da categoria de material para proporcionar uma maior flexibilidade na criação de planos de trabalho, podendo estes ser genéricos e comuns para várias reparações semelhantes. Deste modo, o planeamento previamente desenvolvido, no qual reparações expresso tinham um plano próprio, podia ser também respondido.

Além do exposto, seria necessário o aperfeiçoamento da implementação apresentada atualmente, nomeadamente numa interface gráfico com o colaborador mais intuitiva e um sistema de suporte de base de dados, para guardar as informações adquiridas durante o uso do sistema.

Anexos

Anexo 1 - Cenários

Cenário 1

O cliente dirige-se ao estabelecimento com um equipamento avariado e pede um orçamento para uma reparação. O funcionário do balcão, após se ter autenticado, regista a entrega do equipamento pelo cliente (utilizando o NIF deste para o identificar), bem como o pedido de orçamento. Após o técnico de reparações fazer o orçamento, este é enviado por email ao cliente. O cliente responde, também por email, confirmando o pedido de reparação. Quando o técnico regista a conclusão da reparação é enviada nova notificação por email ao cliente. Este vai à loja levantar o equipamento e o funcionário do balcão regista a entrega do mesmo e o pagamento.

Variantes:

1. Se cliente não confirmar a reparação, no espaço de 30 dias, o orçamento é arquivado.
2. Se o cliente recusar o orçamento, terá que recolher o equipamento.
3. Se o cliente não vier levantar o equipamento, no espaço de 90 dias, este vai para uma lista de equipamento abandonado. Nessa altura, pode ser dada baixa do equipamento.

Cenário 2

O cliente dirige-se ao estabelecimento com um equipamento para ser realizado um Serviço ExpressoTM. Os serviço expresso são intervenções a preço fixo que o centro realiza (substituir um écran de telemóvel, instalar um sistema operativo, etc.). O funcionário do balcão certifica-se que existe disponibilidade para realizar o serviço de imediato e efectua o seu registo. Quando o técnico assinala a conclusão do serviço, o cliente é notificado por SMS.

Variantes:

1. Caso não exista disponibilidade de tempo para realizar o serviço de imediato sem

atrasar os serviços programados, o serviço expresso é recusado

Cenário 3

O técnico de reparações acede à lista de pedidos de orçamento e escolhe o mais antigo. Utiliza o código de registo do equipamento para o ir buscar ao armazém e, depois de analisar a descrição do problema e o próprio equipamento, regista o plano de trabalhos para a reparação.

O plano de trabalhos consiste na sequência de passos necessária para efectuar a reparação. Para cada passo o técnico define uma previsão do tempo necessário para a sua execução, bem como o custo das peças a utilizar, caso sejam necessárias. Um passo pode ser decomposto em sub-passos, caso em que a sua duração e custo de peças serão a soma das durações e custos de peças dos sub-passos.

A definição do plano de trabalhos permite obter uma previsão do número total de horas de trabalho e o custo das peças necessárias. Com base nessa informação, é criado um orçamento que é enviado ao cliente. Nesse orçamento vai também o prazo máximo de execução da reparação, calculado em função do tempo necessário para reparar o equipamento e o trabalho actualmente por realizar.

Variantes:

1. Se o equipamento não poder ser reparado, essa informação é enviada ao cliente.

Cenário 4

O técnico acede à lista de equipamentos a reparar e escolhe o mais urgente. Vai assinalando a execução dos passos conforme os realiza, indicando as horas gastas e custo das peças efectivamente utilizadas. Após efectuar todos os passos da reparação, regista a sua conclusão.

Variantes:

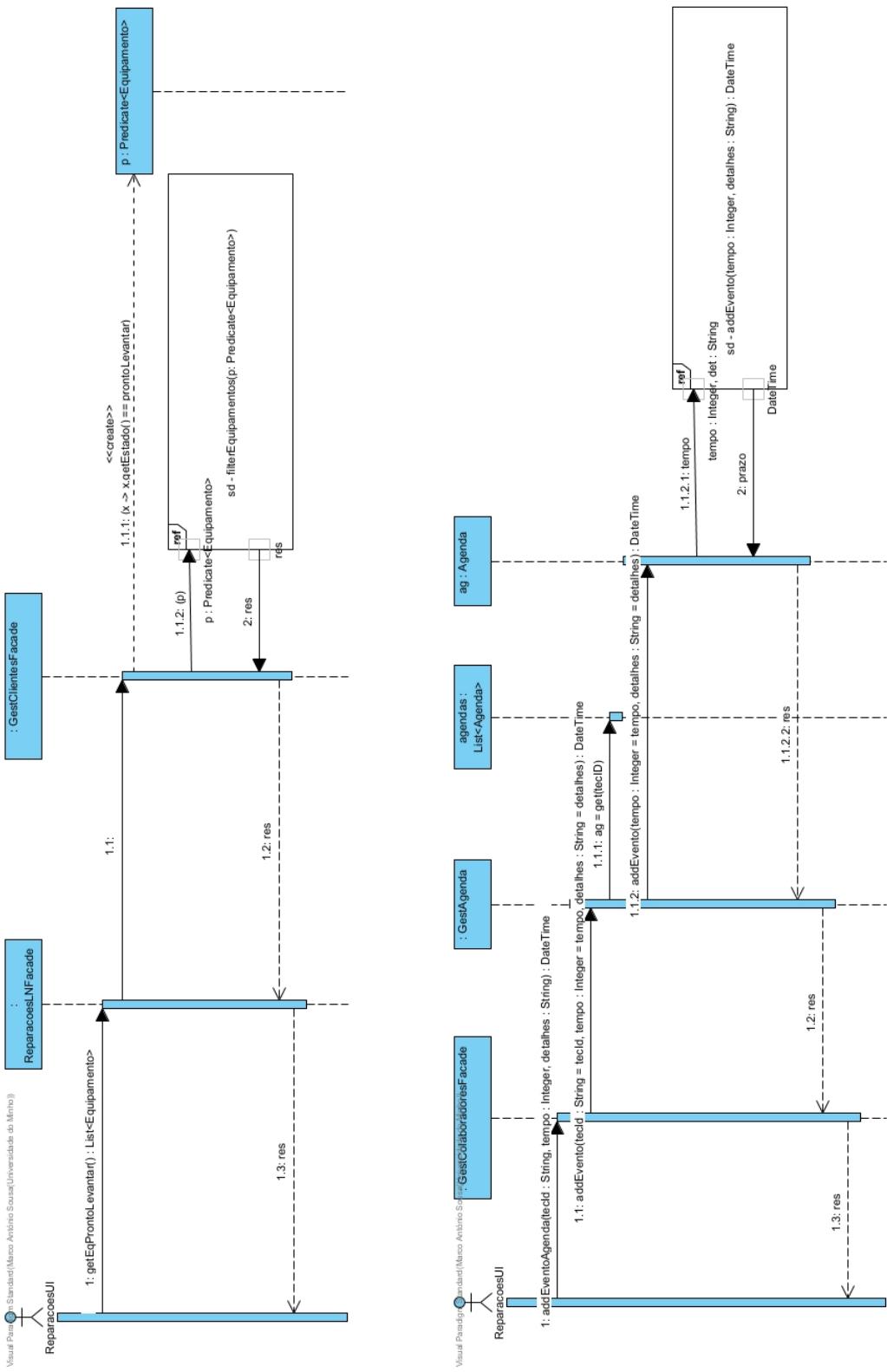
1. Caso o técnico necessite interromper a reparação (por falta de tempo ou peças) pode colocar a reparação em espera.
2. Se, durante a reparação, se concluir concluir que o custo final da reparação irá ser superior a 120€ cliente. Fica registada a data e hora do contacto e quem o realizou. O cliente pode aceitar ou não a reparação
3. Se se tratar de um serviço expresso não existe um plano de trabalhos associado à reparação, pelo que o técnico se limita a indicar a conclusão do serviço.

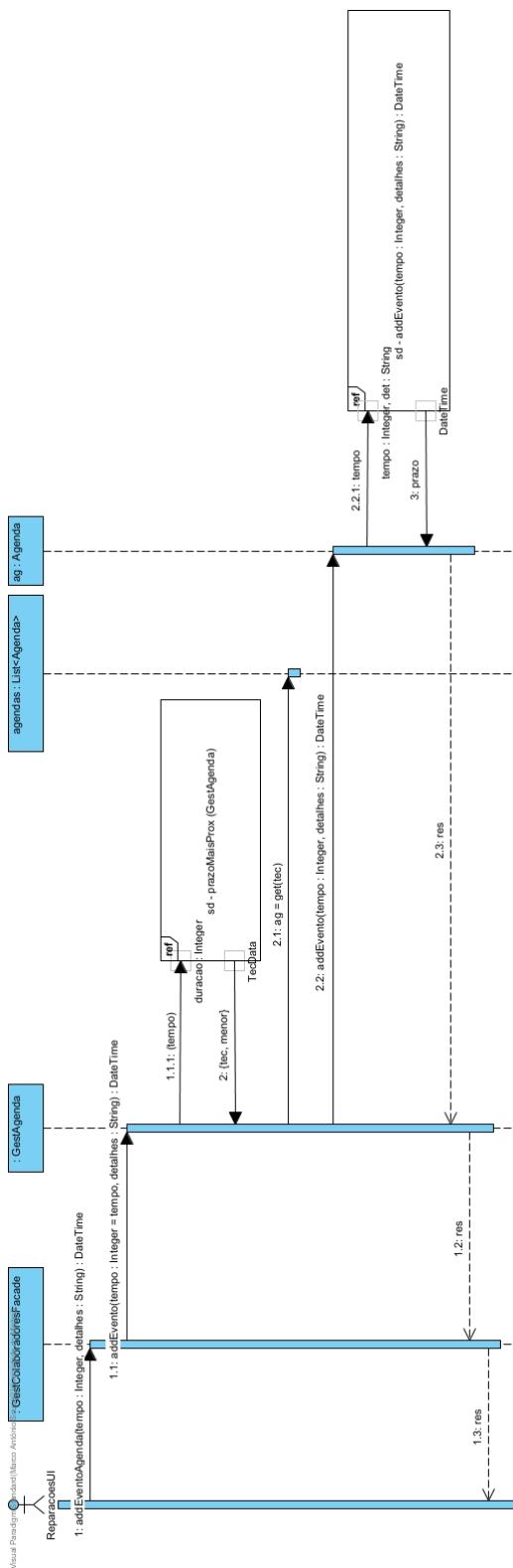
Cenário 5

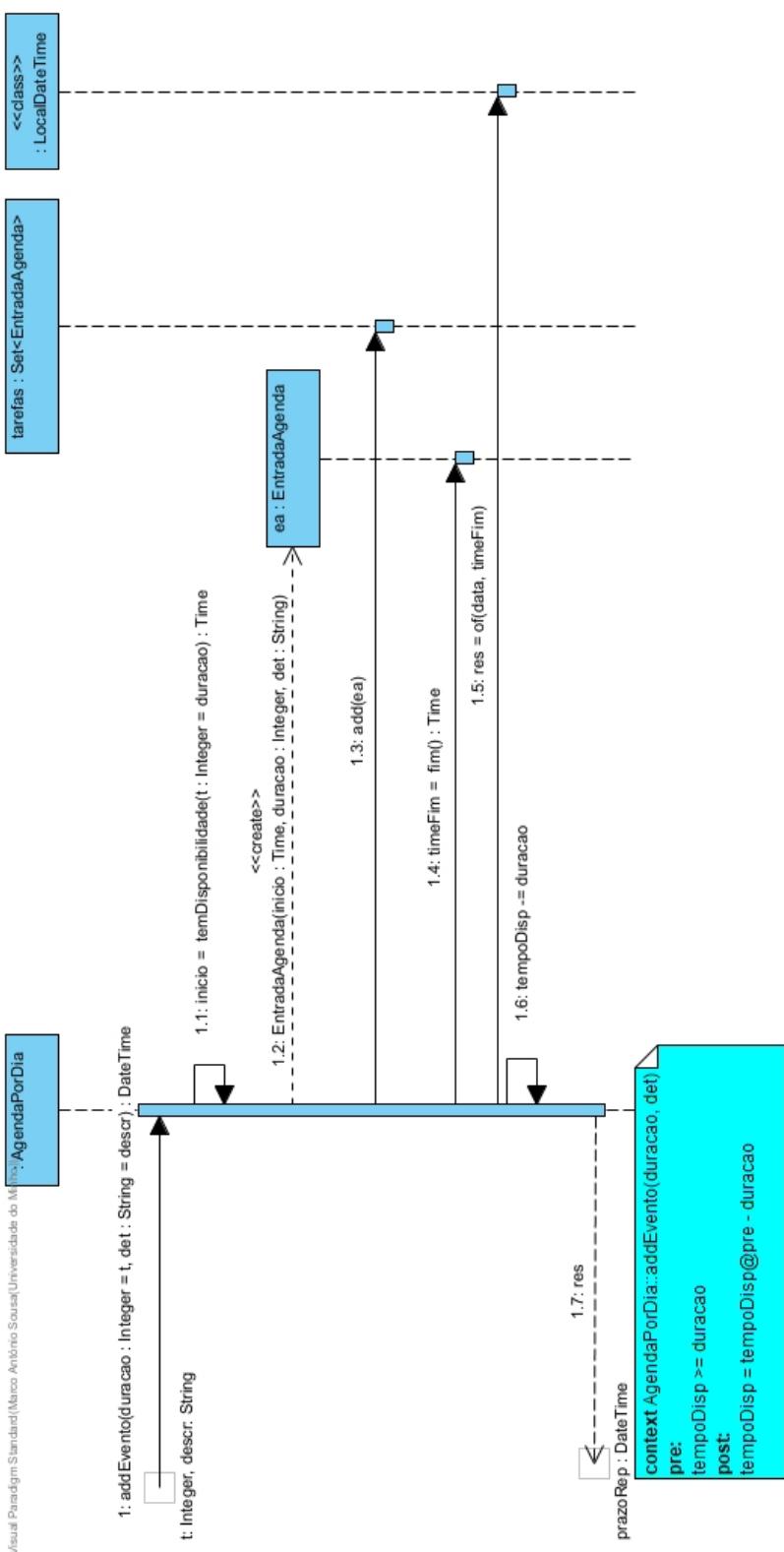
Chegado ao final do mês, e para avaliar o funcionamento do centro, o gestor do centro consulta três tipos de listagem

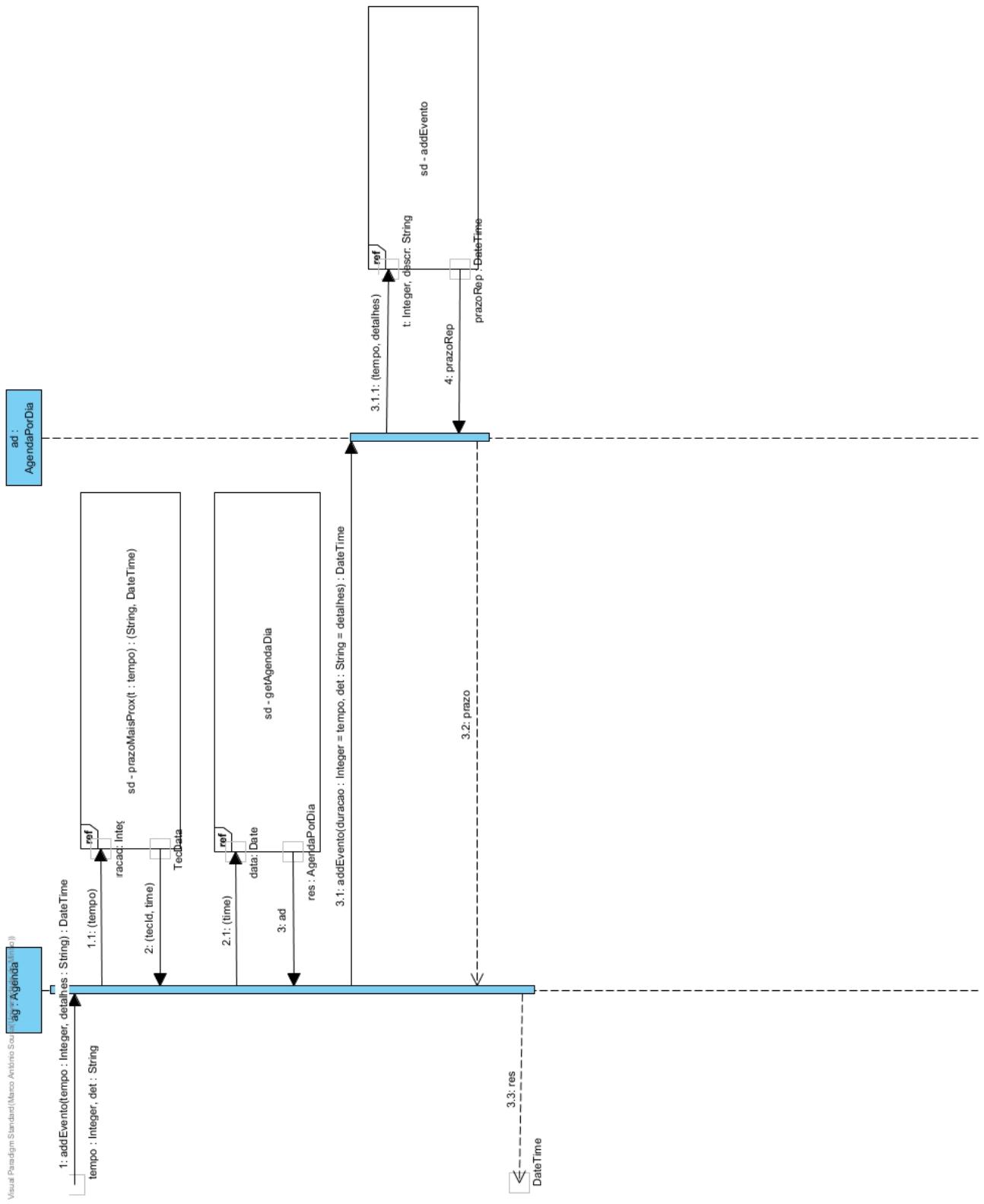
- uma listagem em que para cada técnico de reparações é indicado o número de reparações programadas/expresso realizadas, a duração média das reparações programadas realizadas e a média dos desvio em relação às durações previstas;
- uma listagem que indica, para cada funcionário de balcão, quantas recepções e entregas de equipamentos realizou;
- uma listagem exaustiva, para cada técnico, de todas as intervenções (passos de reparação e reparações expresso) realizadas.

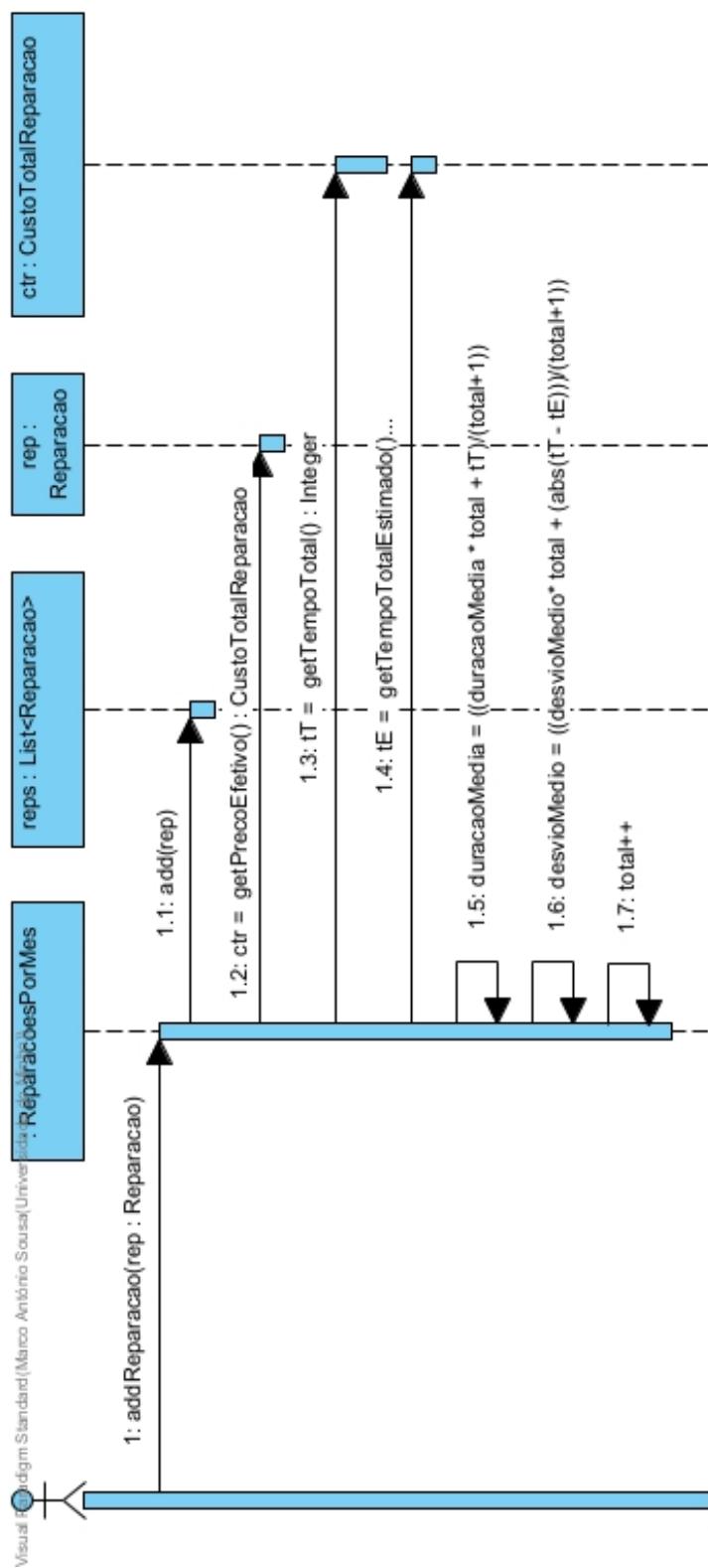
Diagramas de Sequência

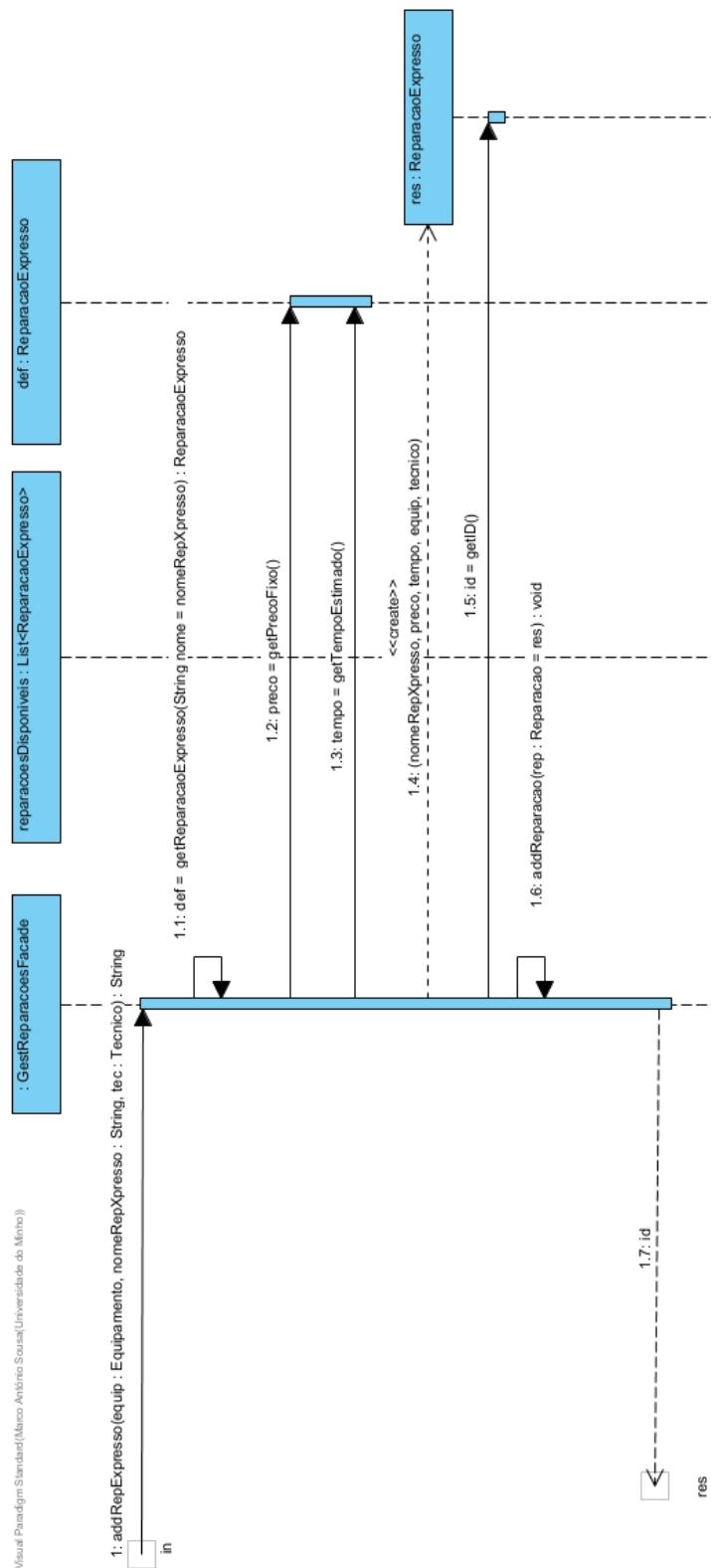


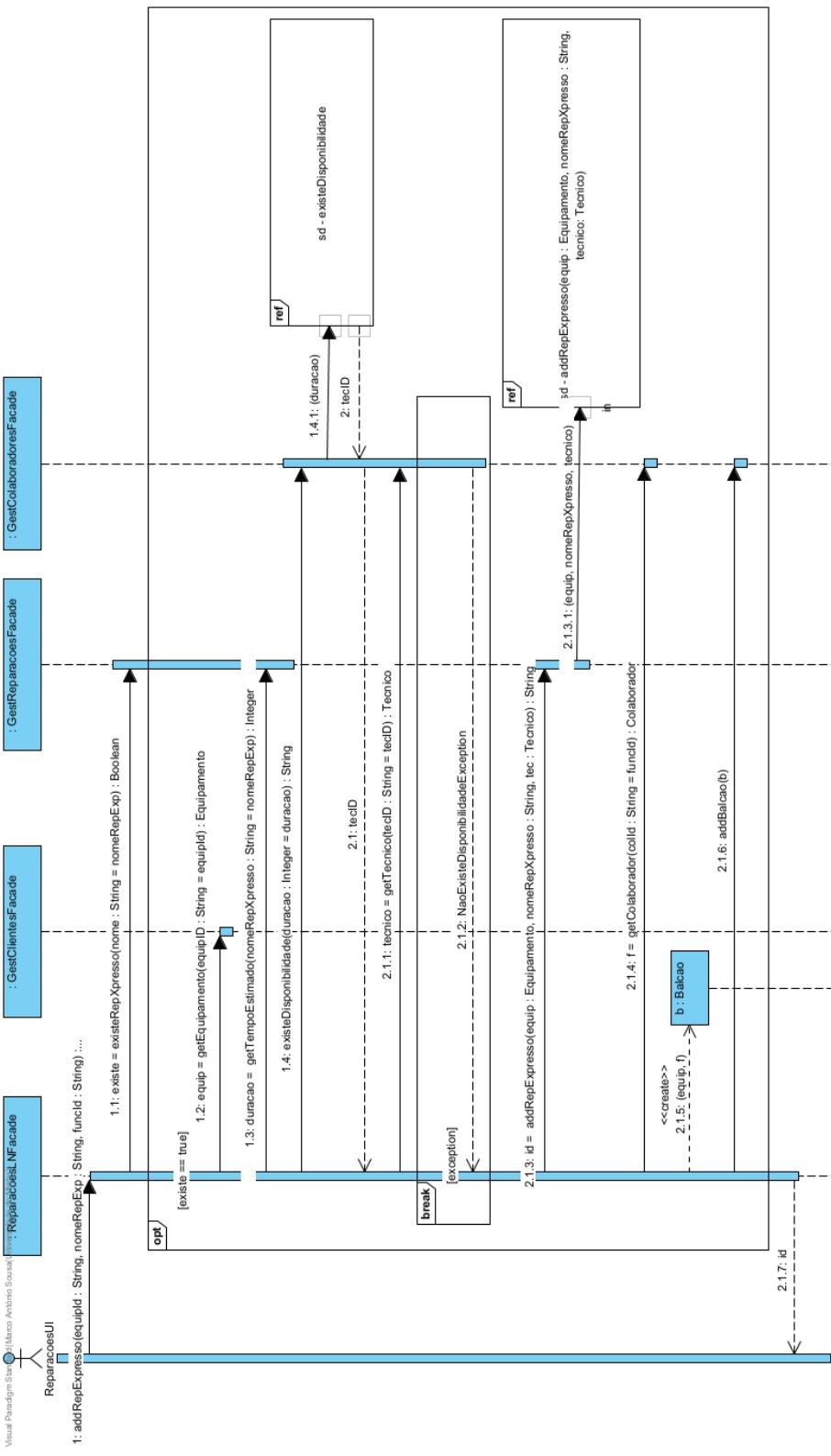


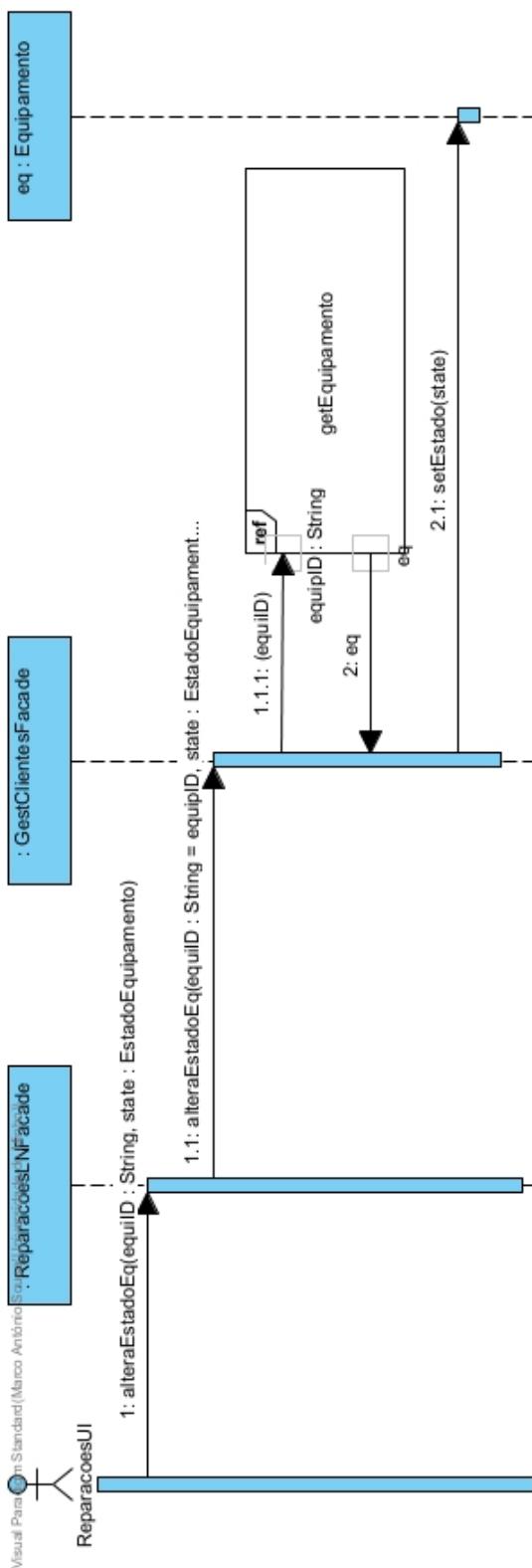


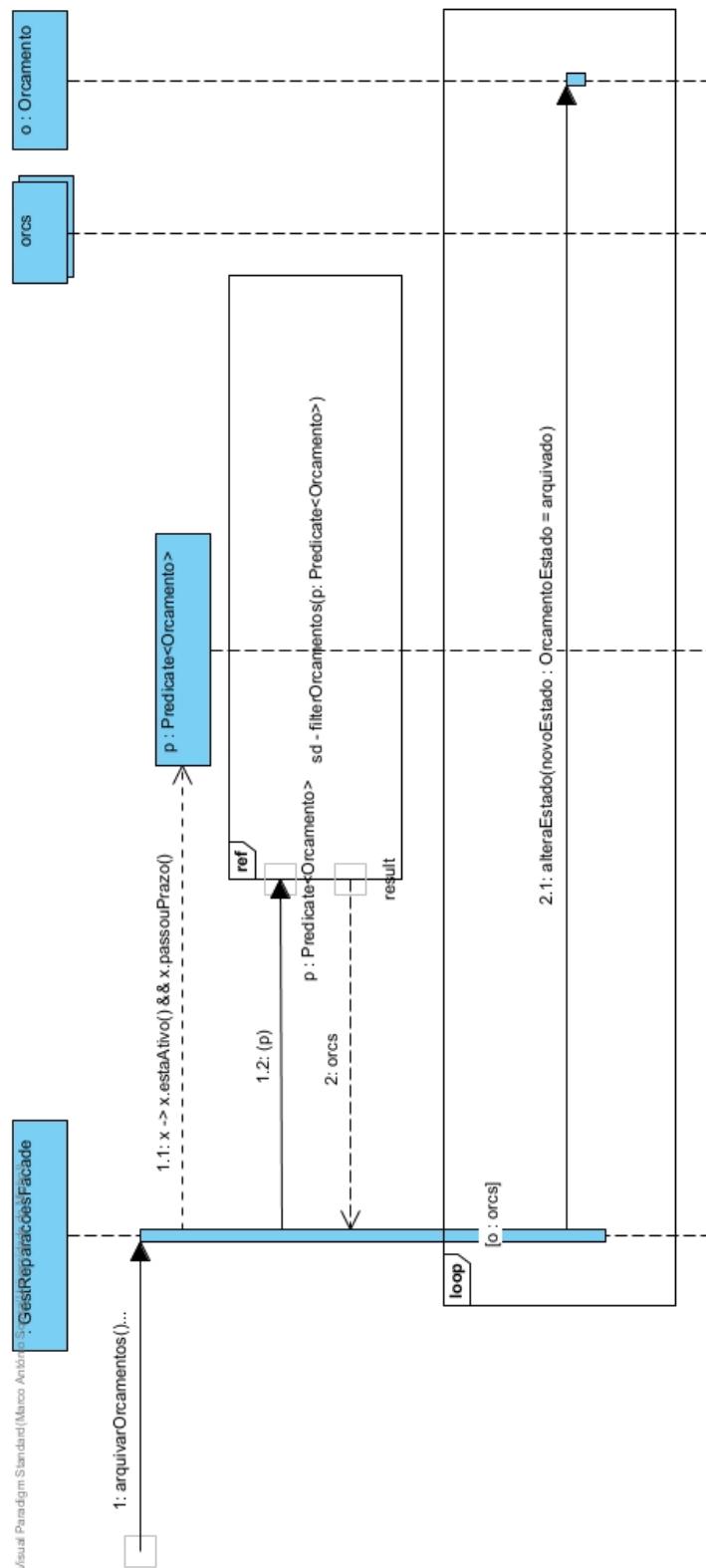


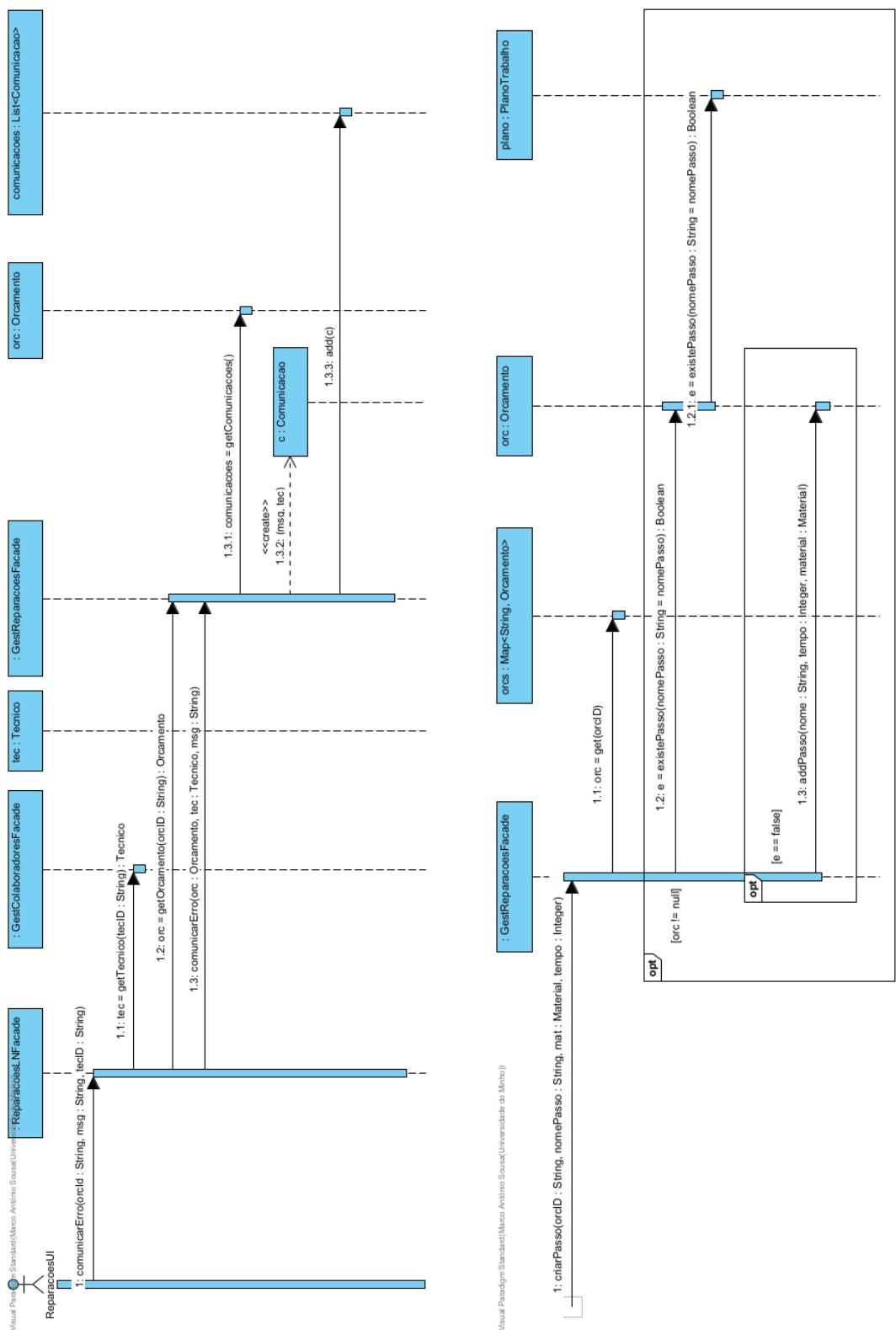


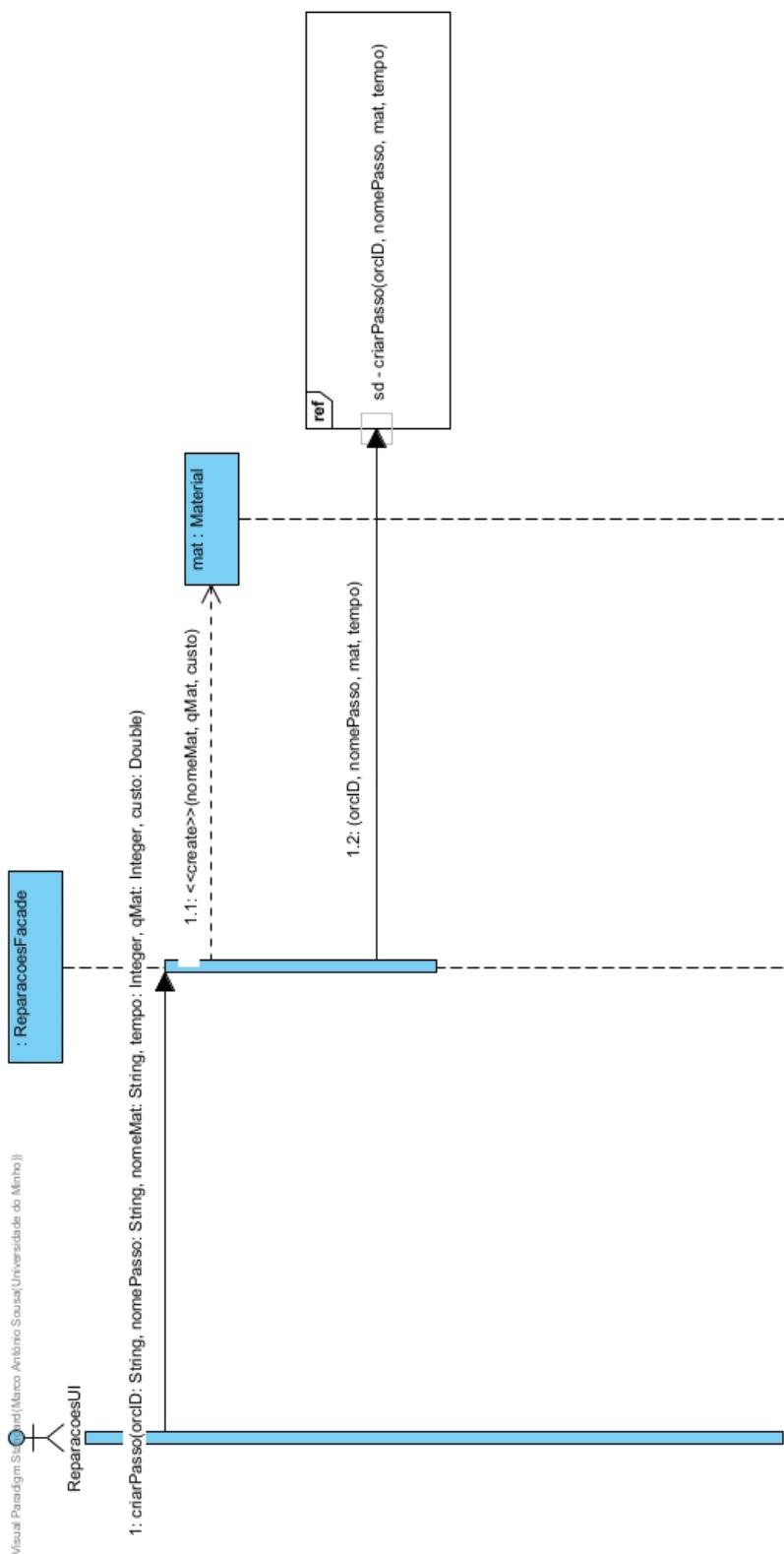


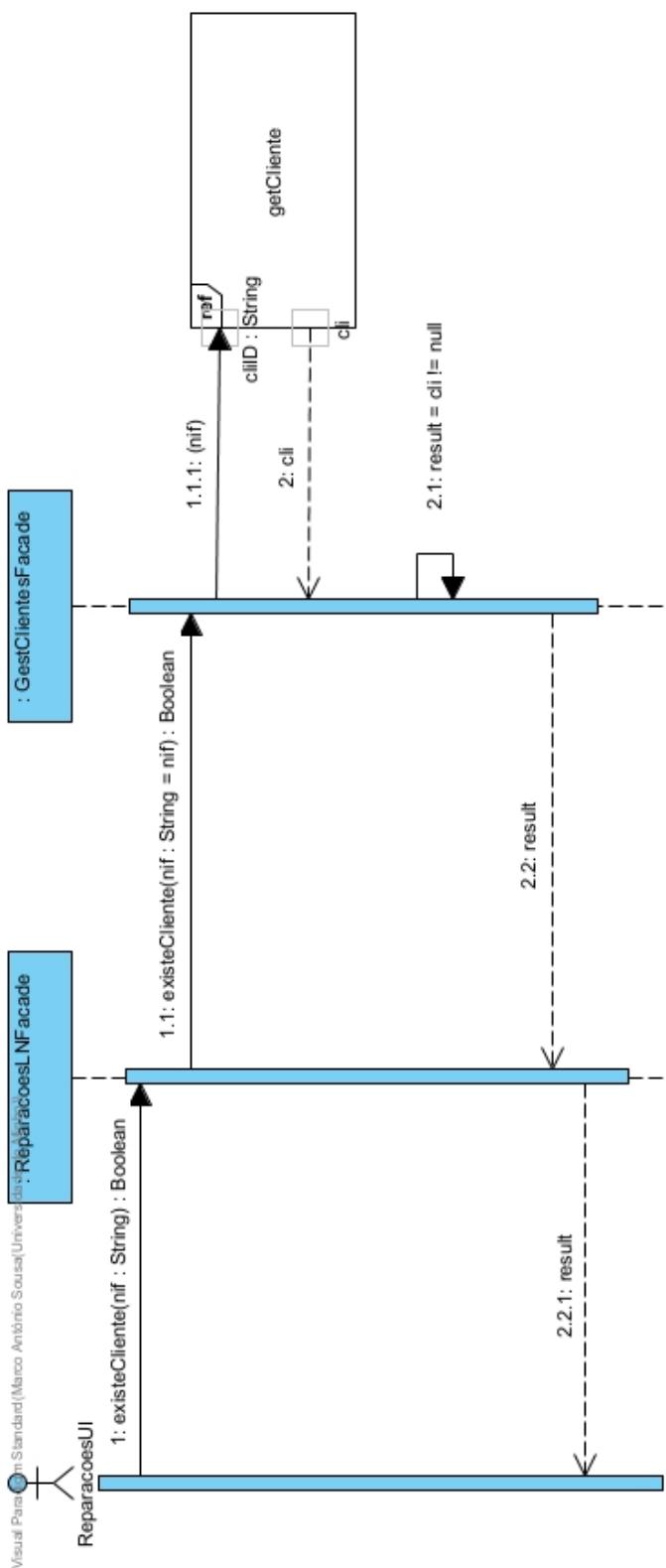


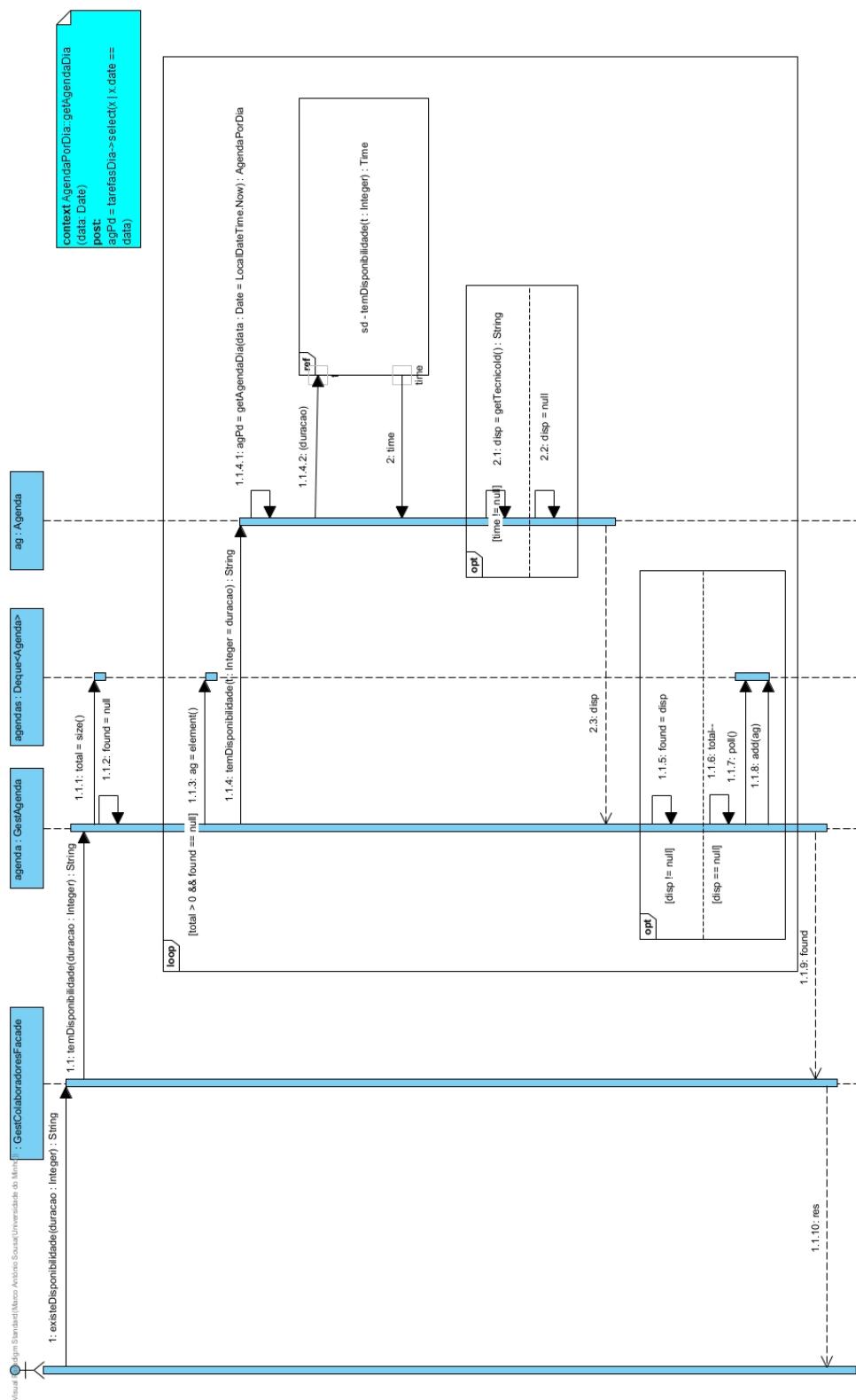


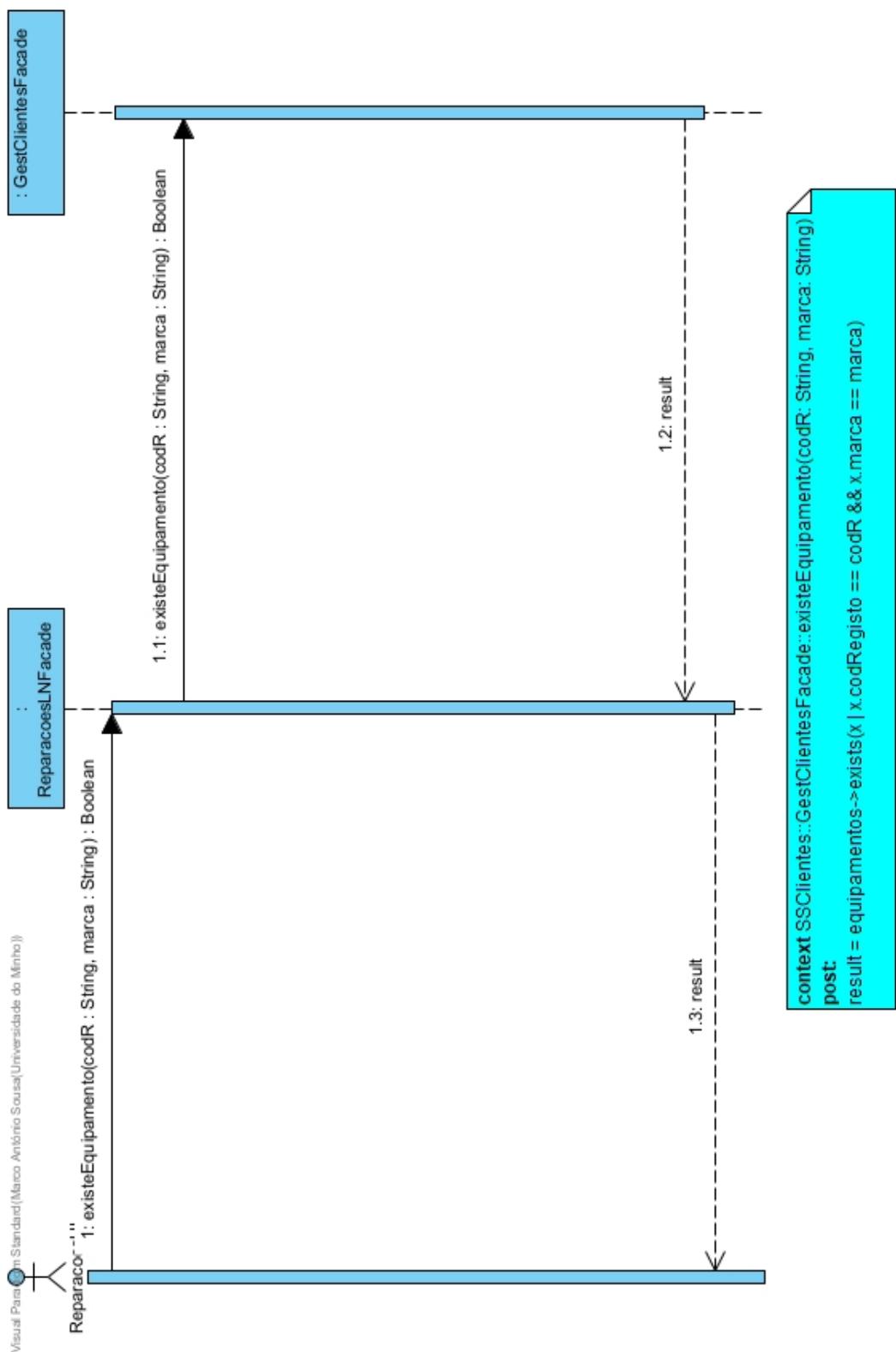


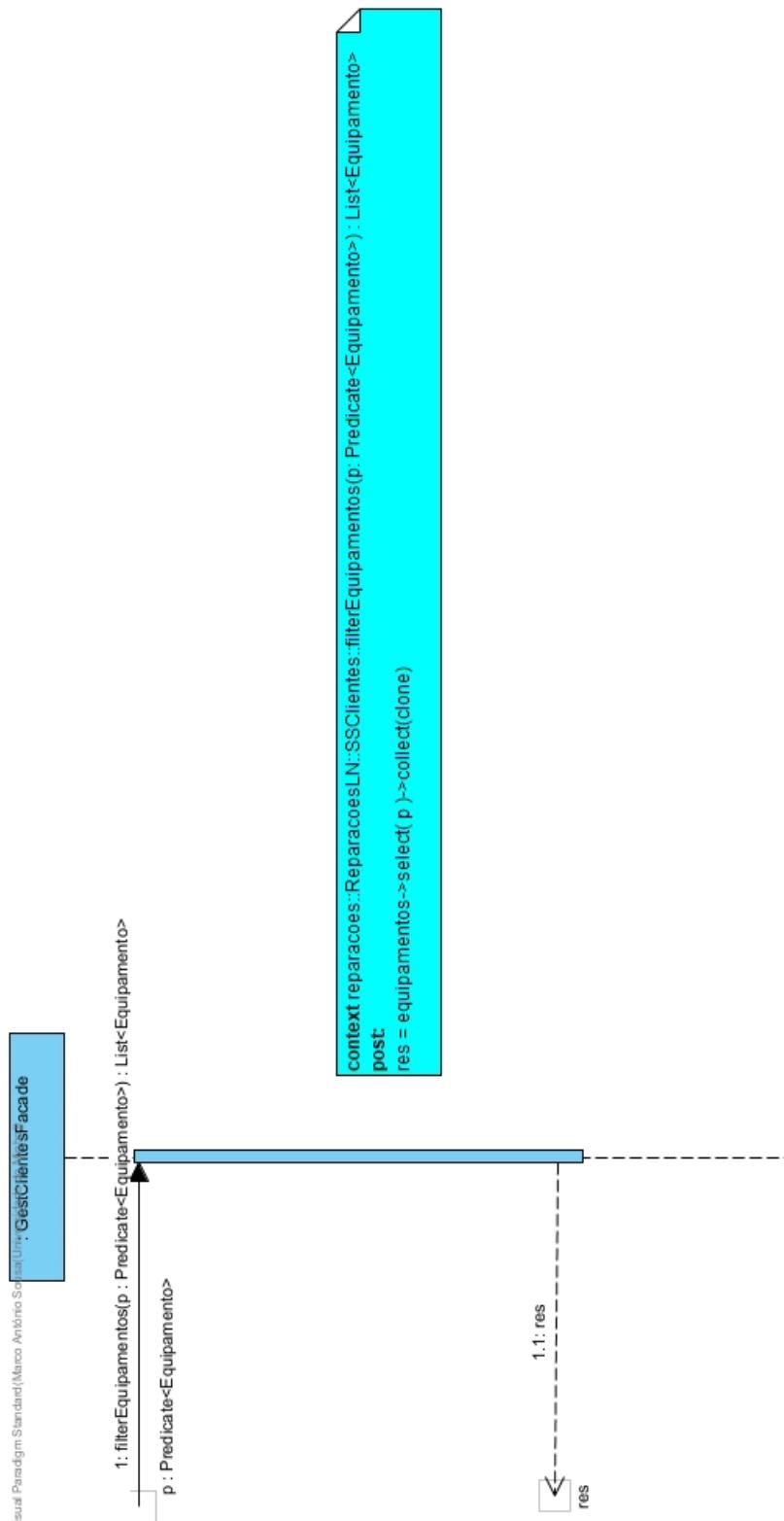


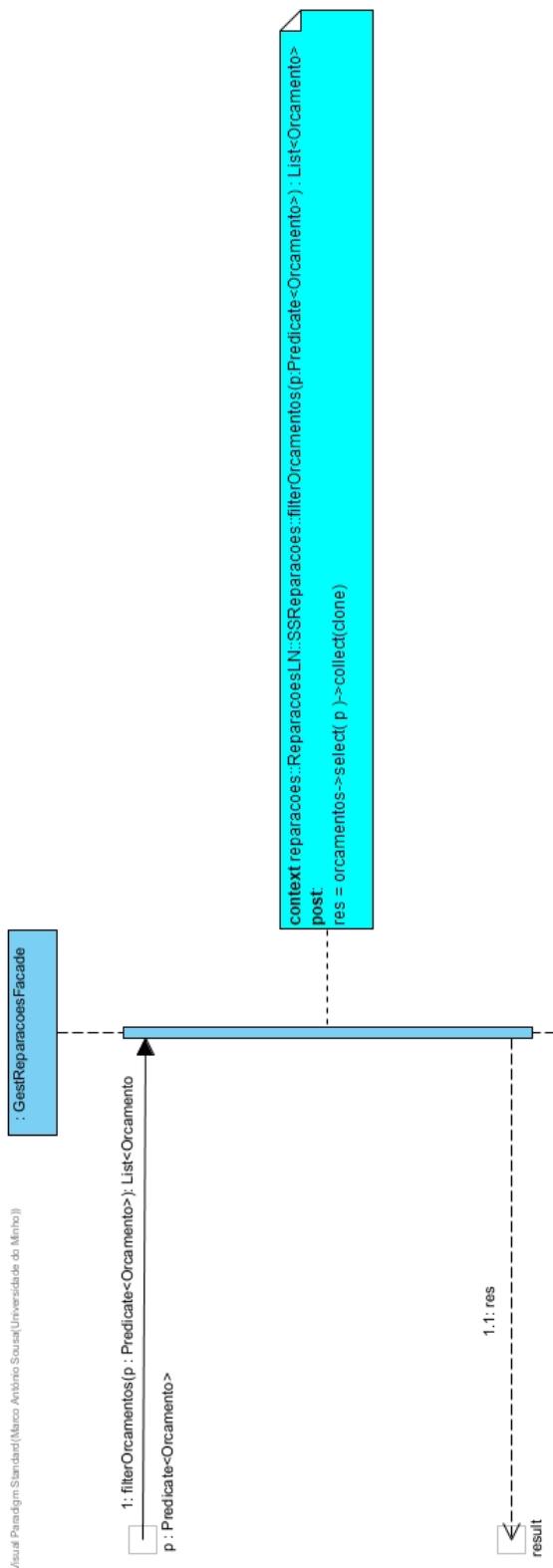


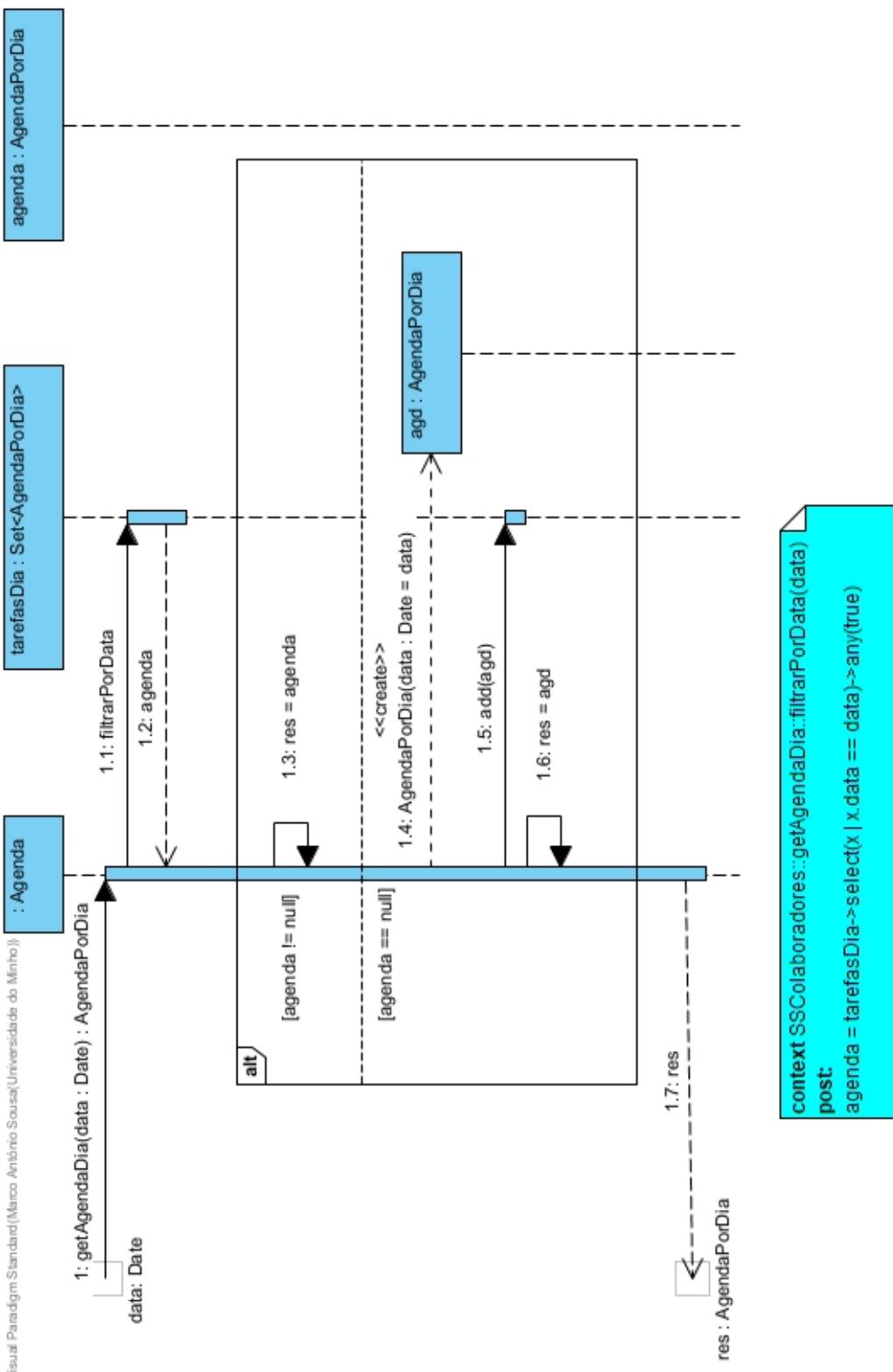


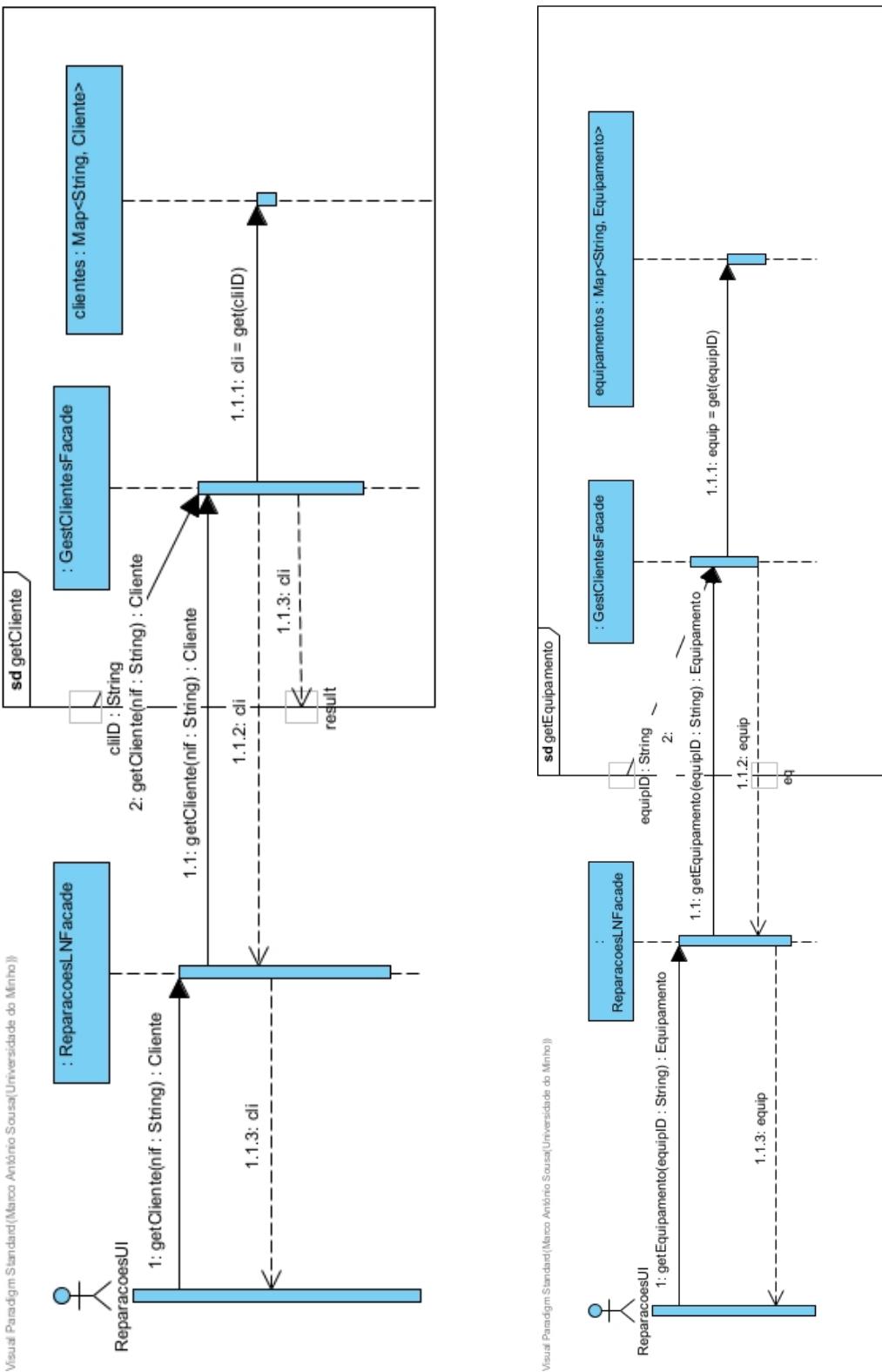


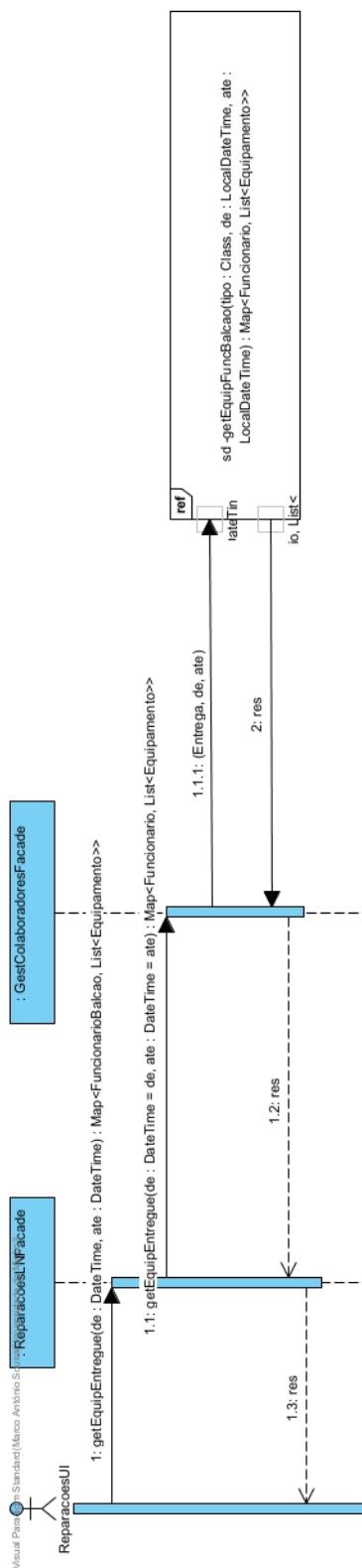


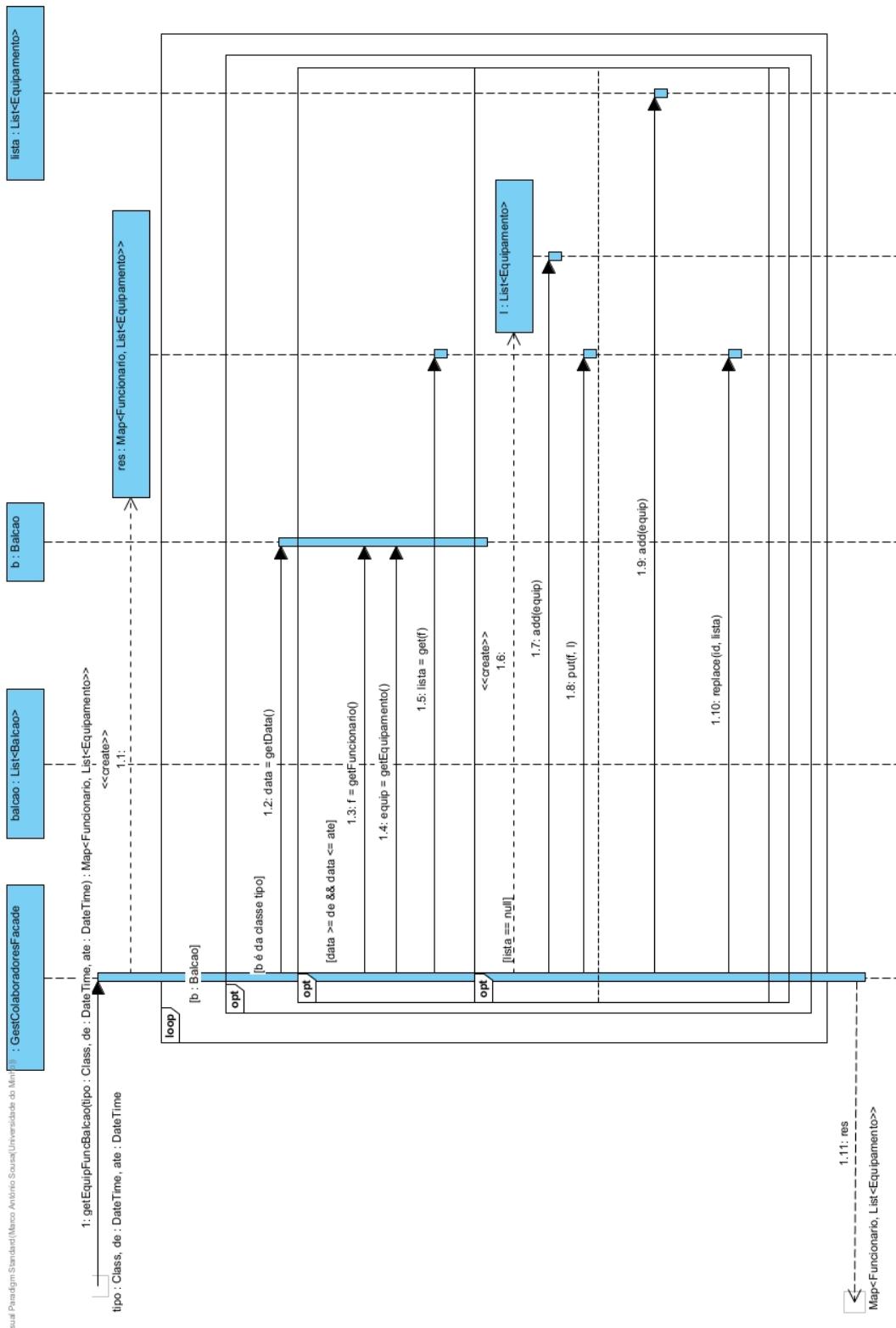


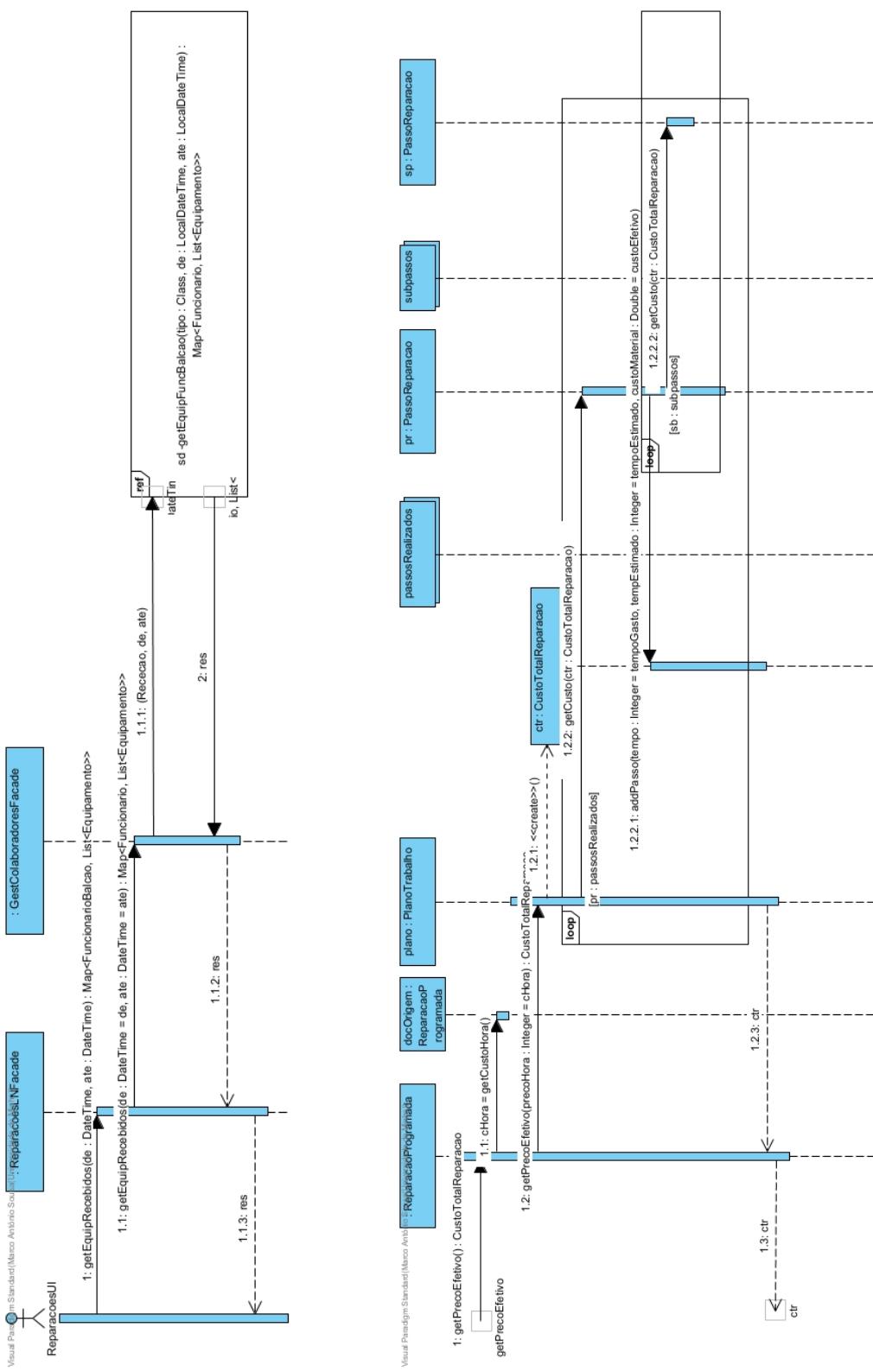


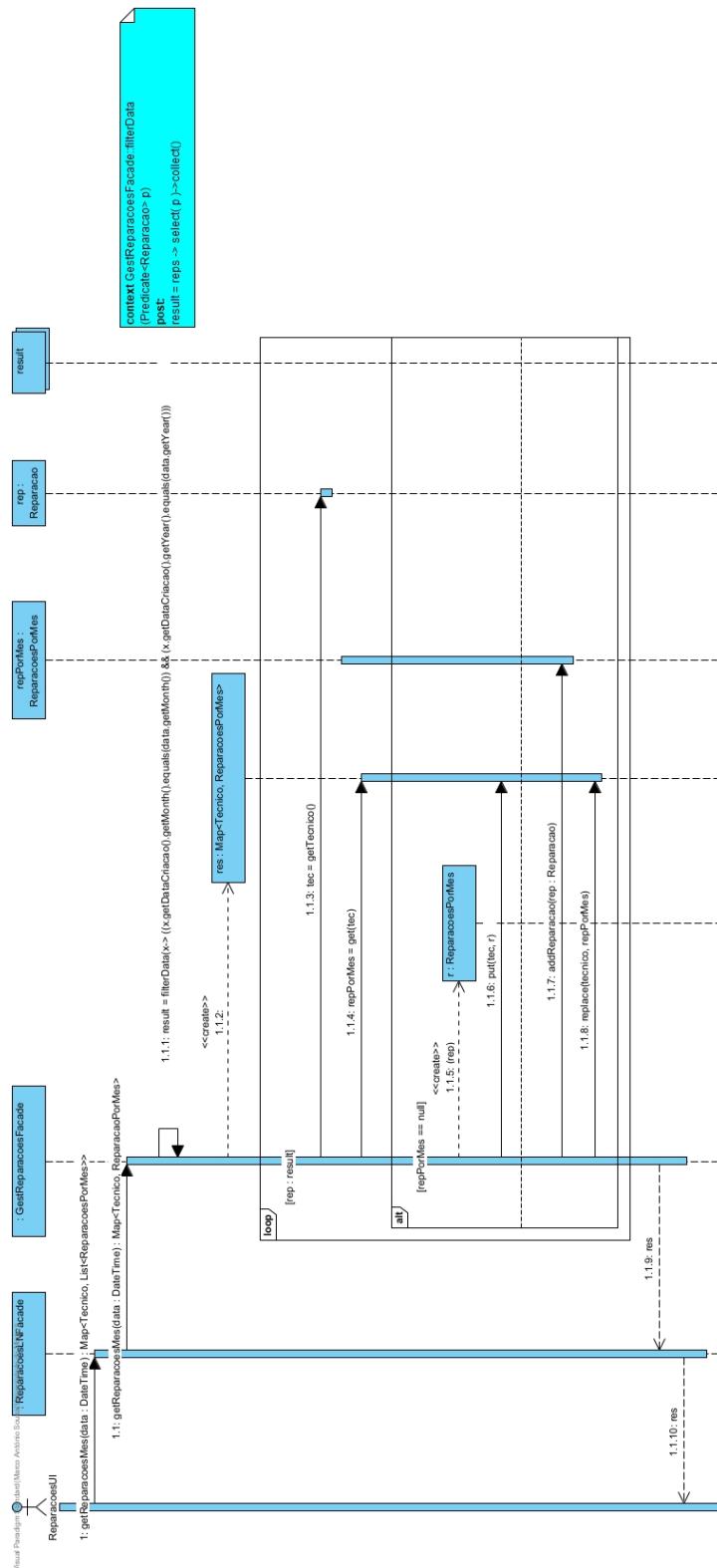


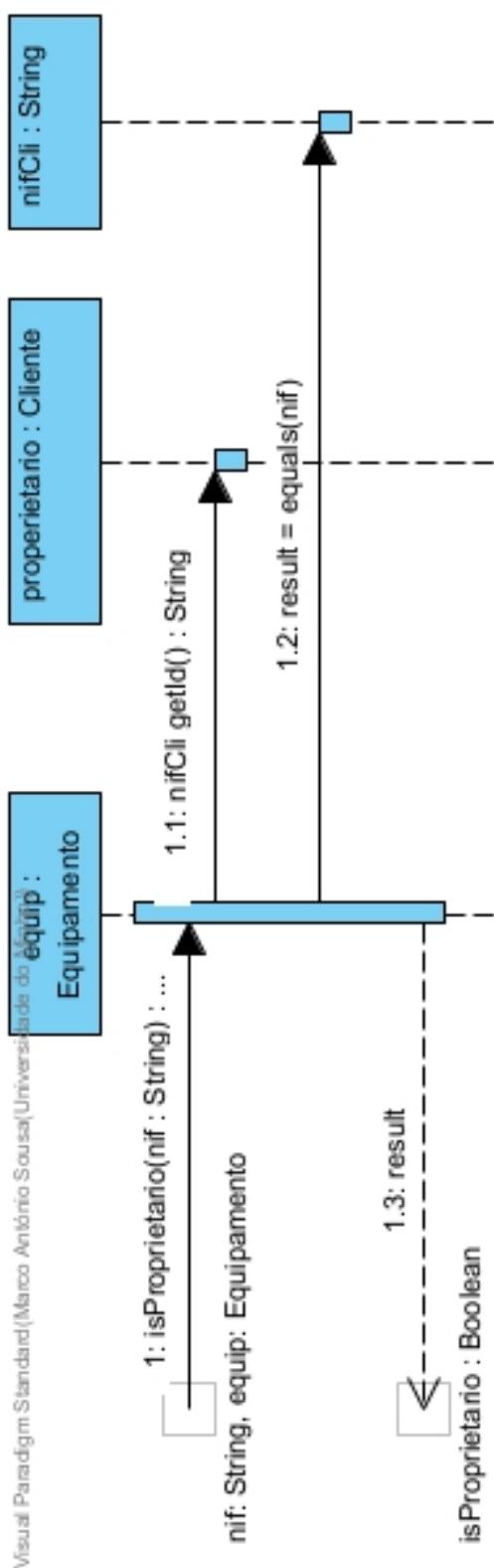


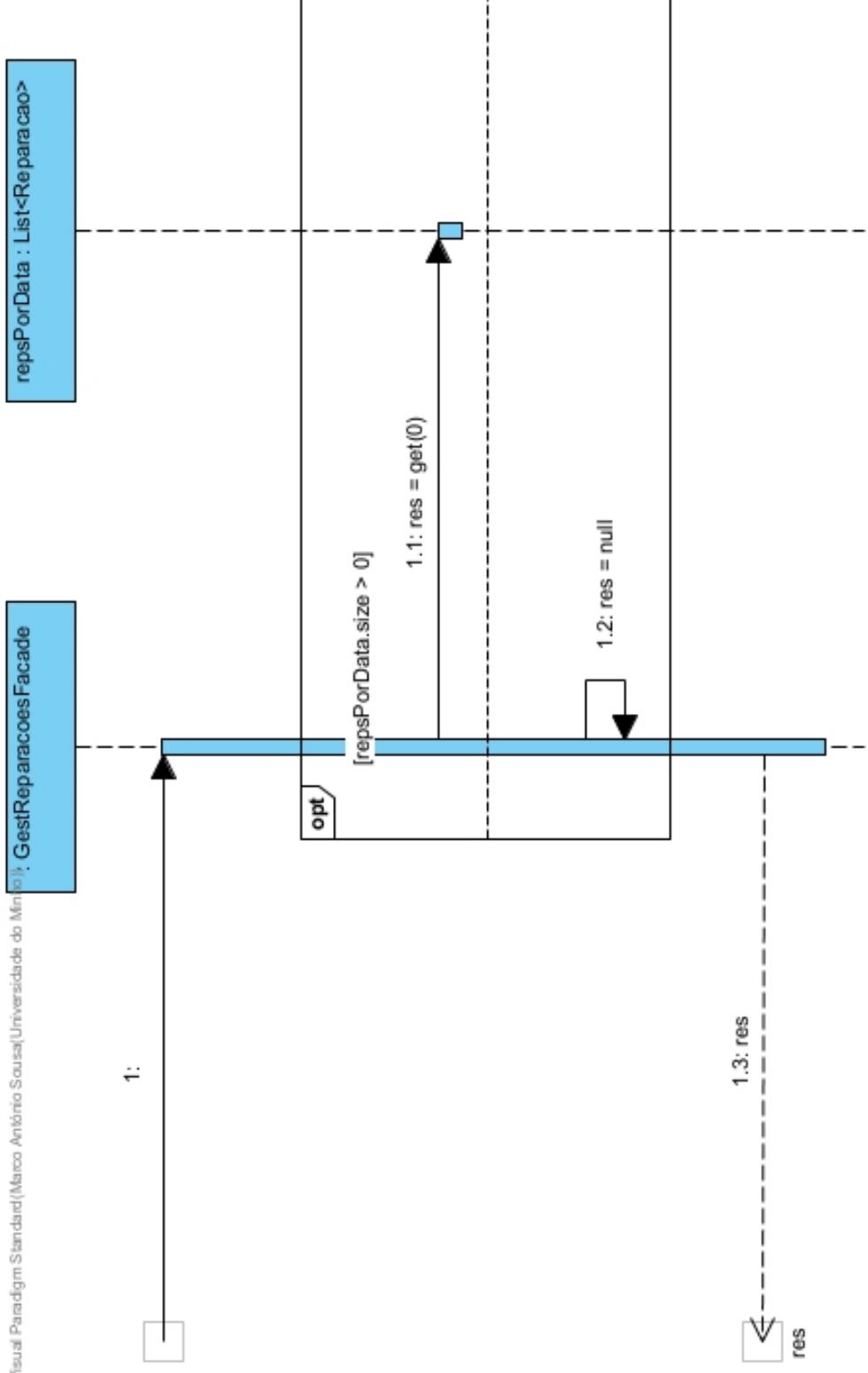


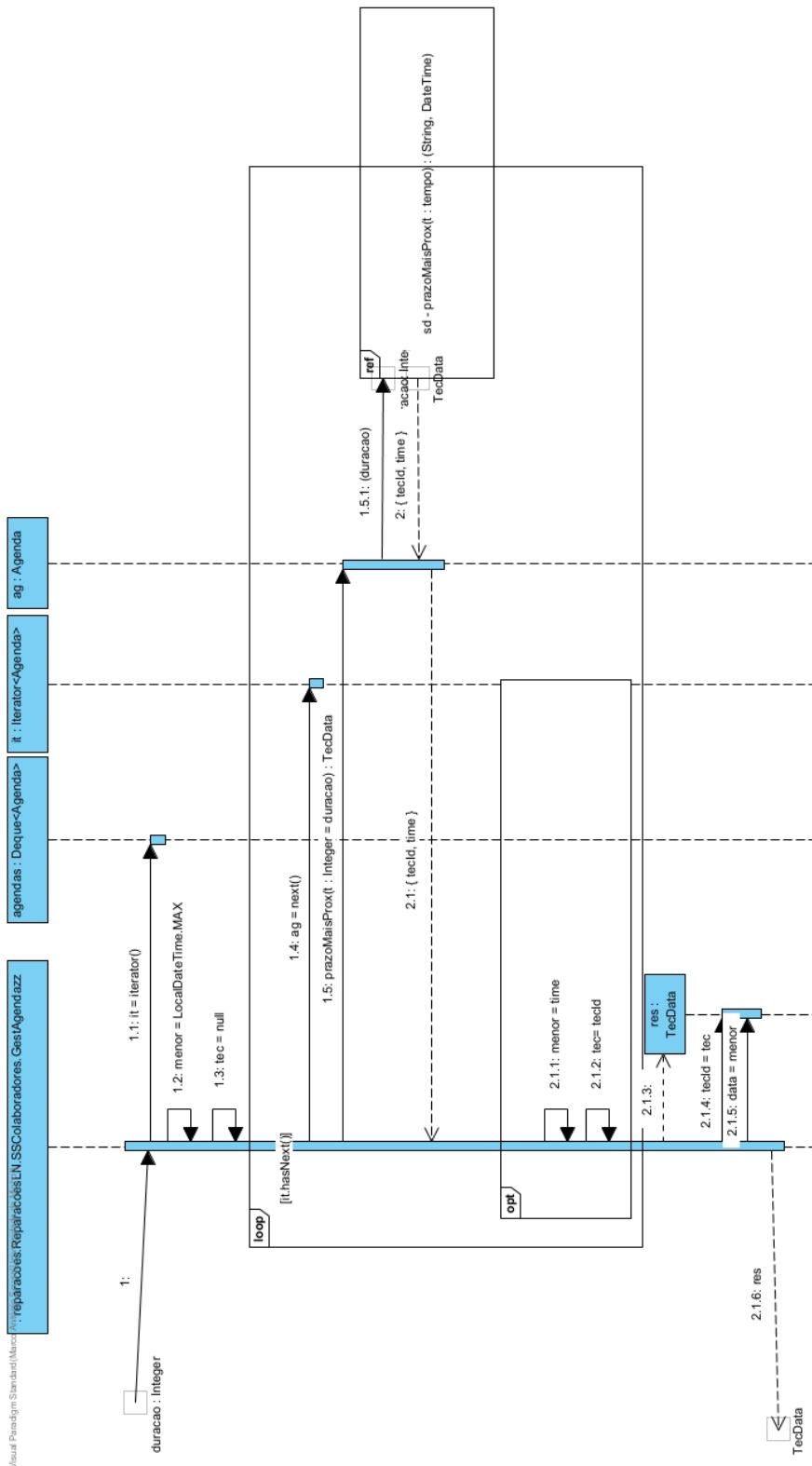


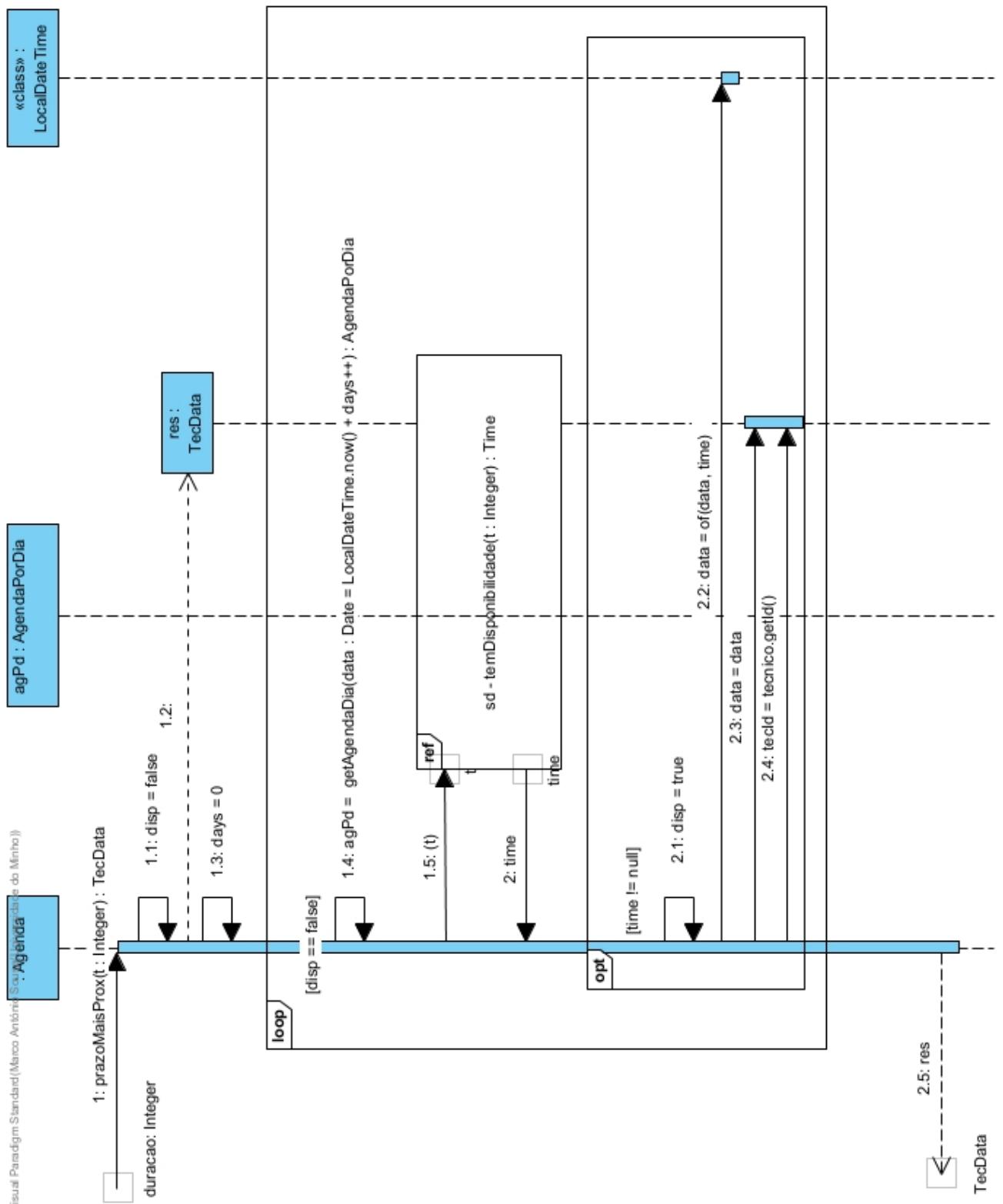


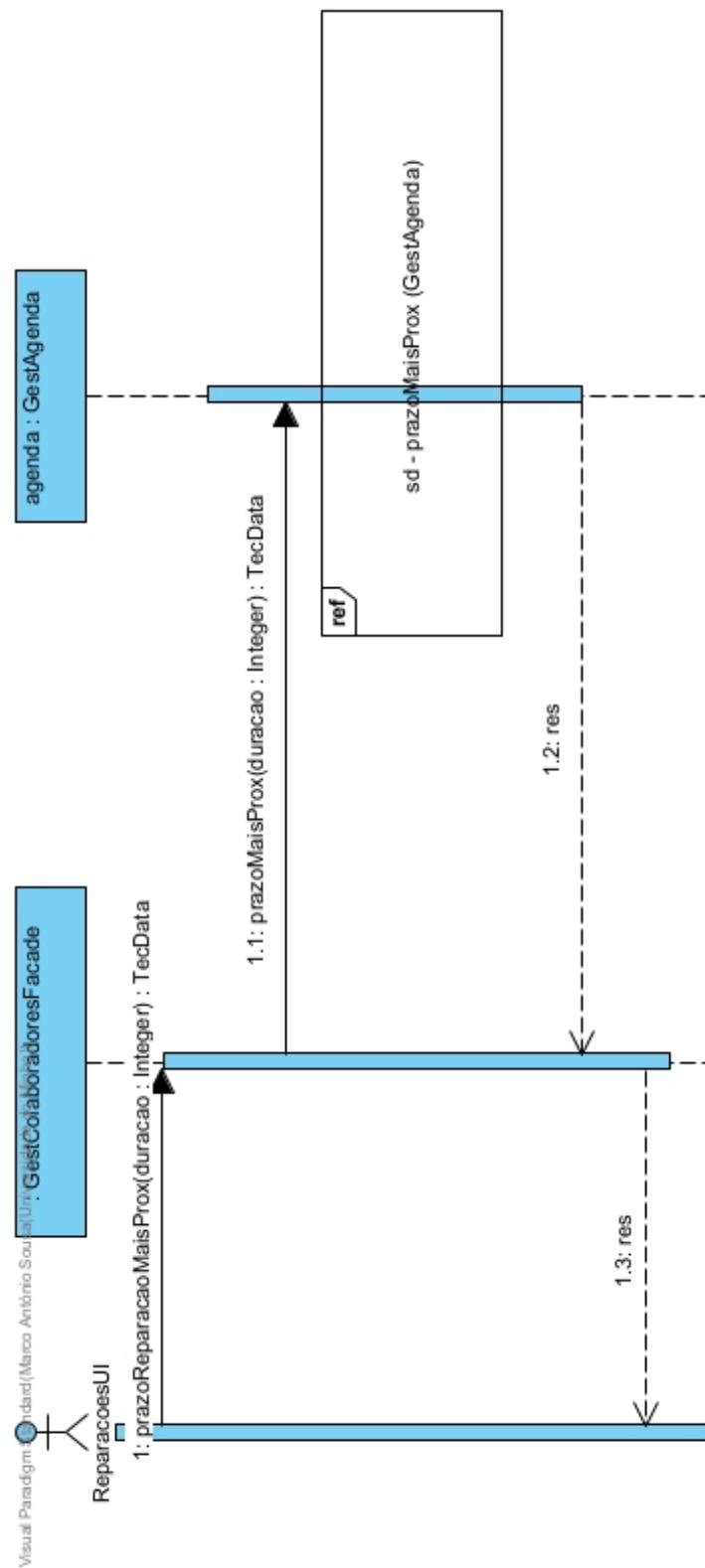


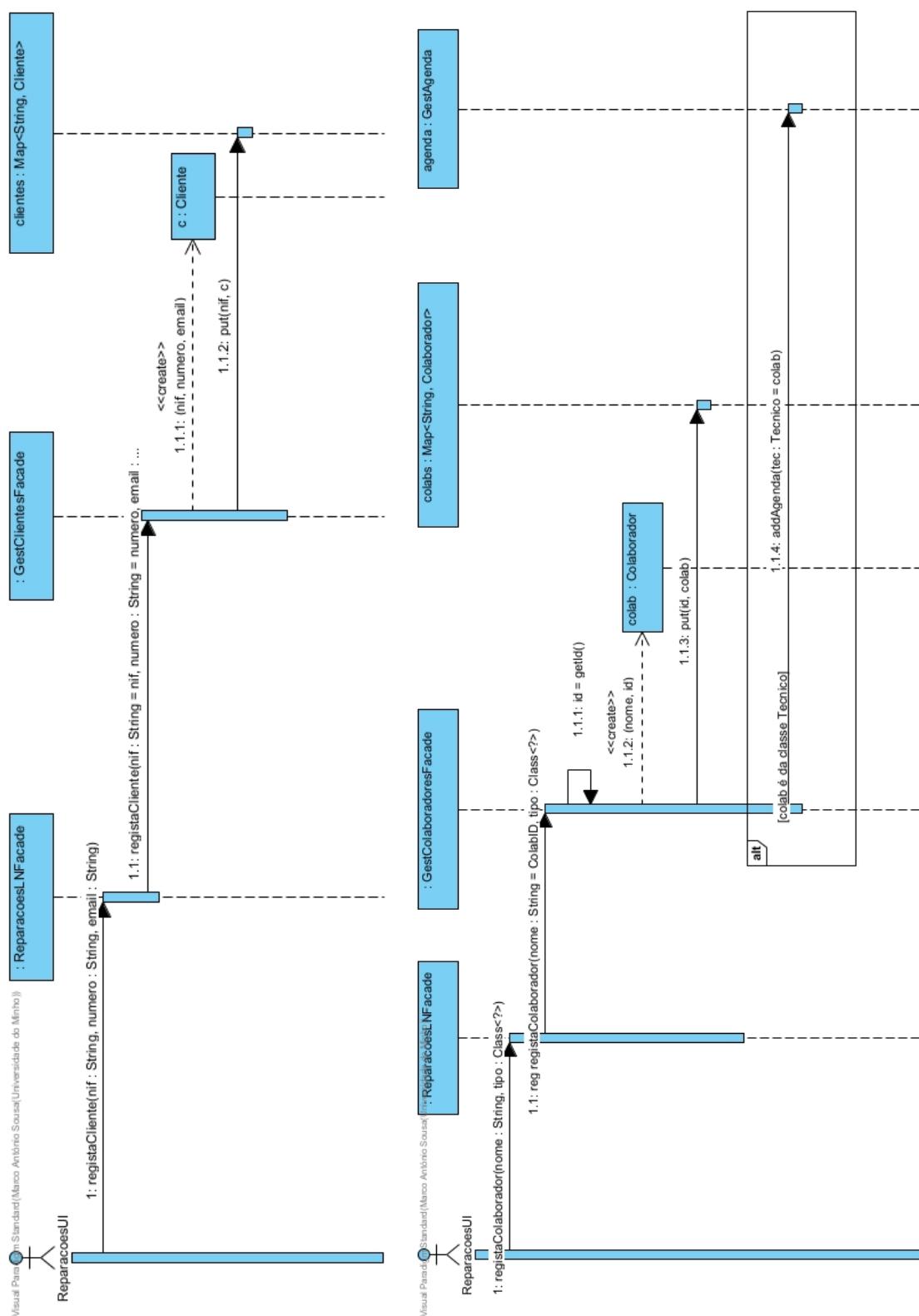


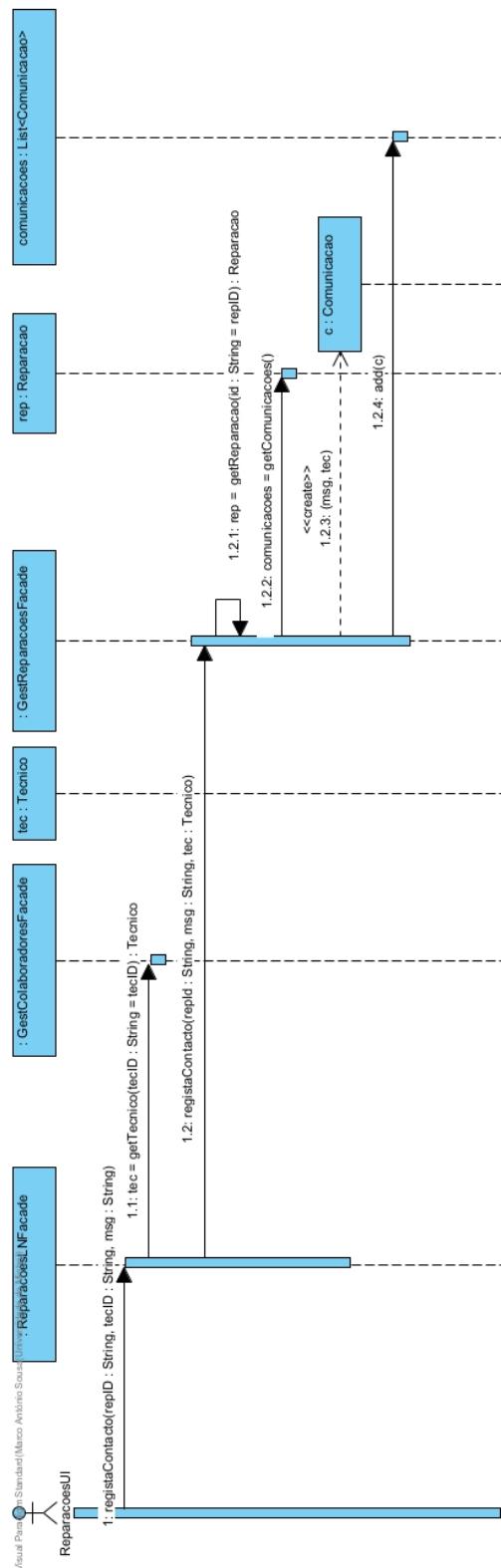


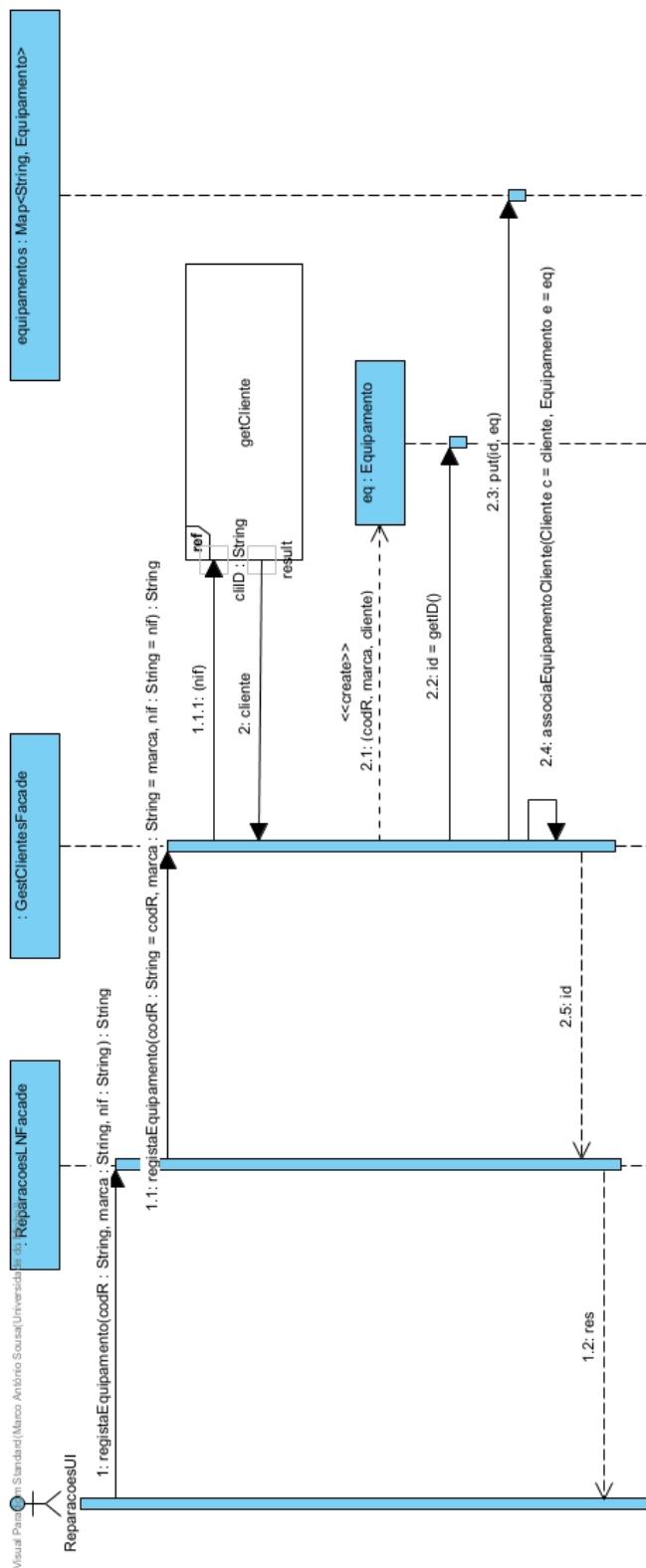


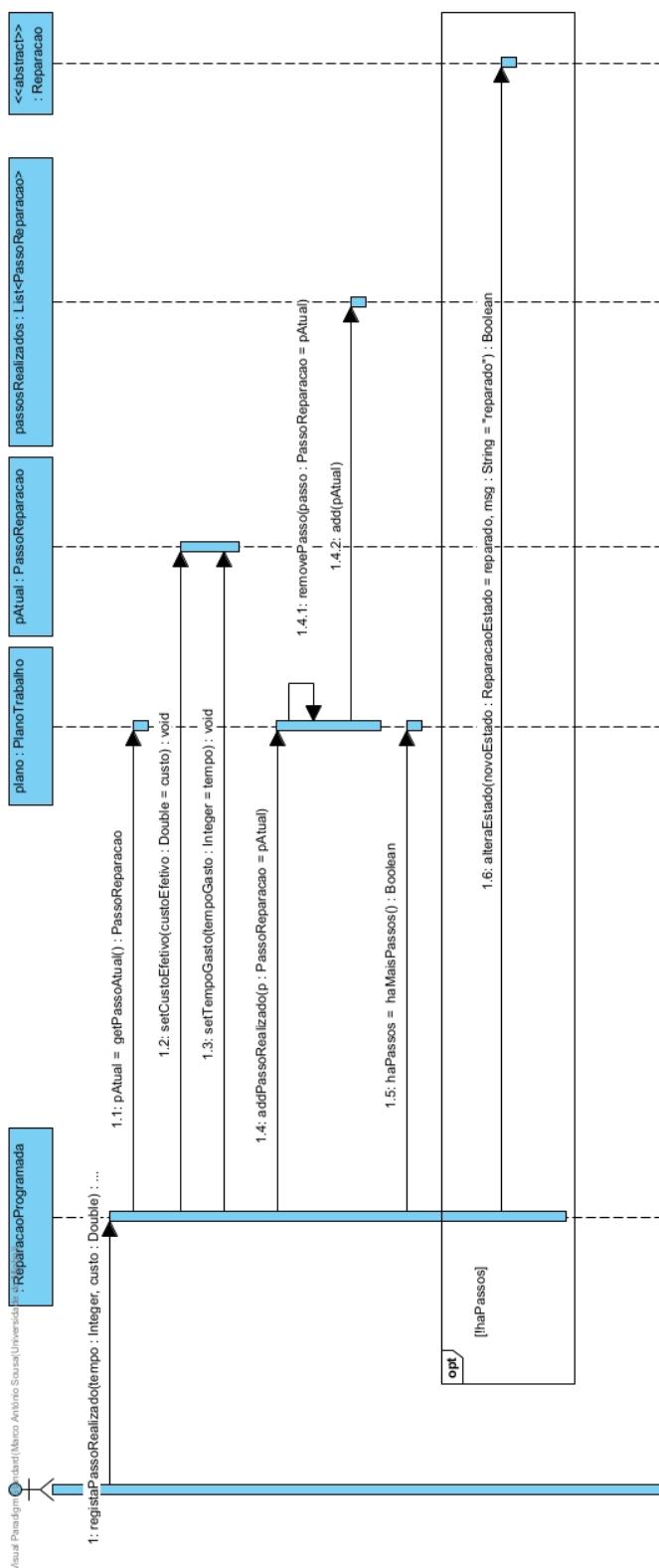


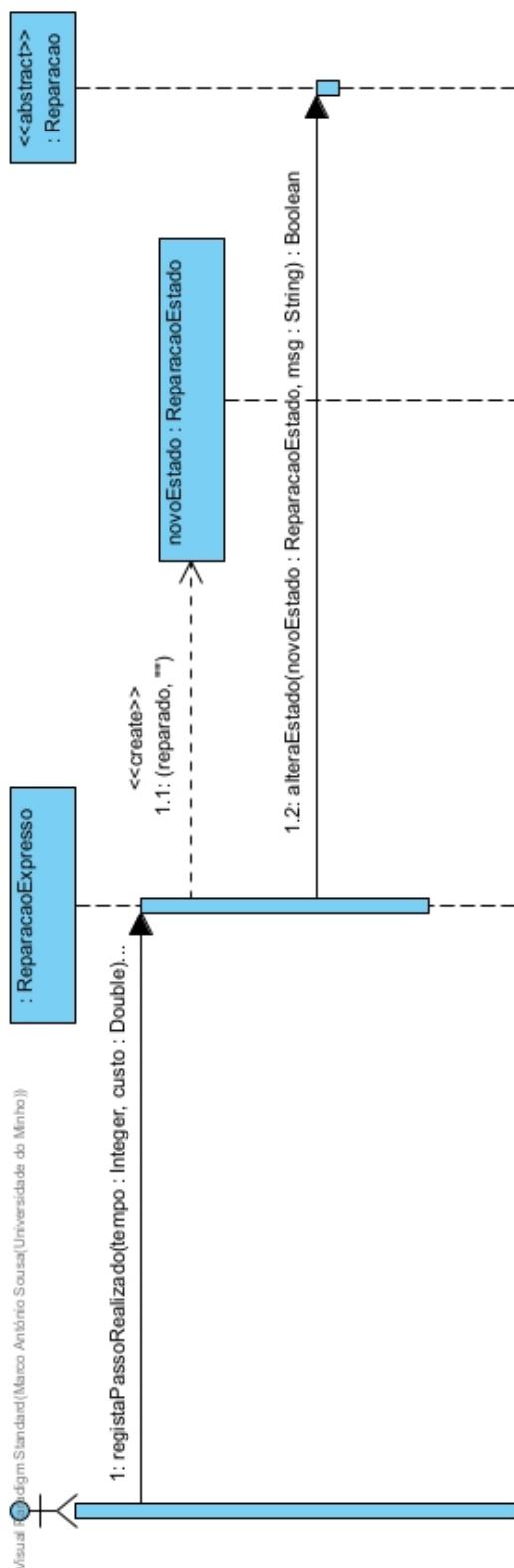


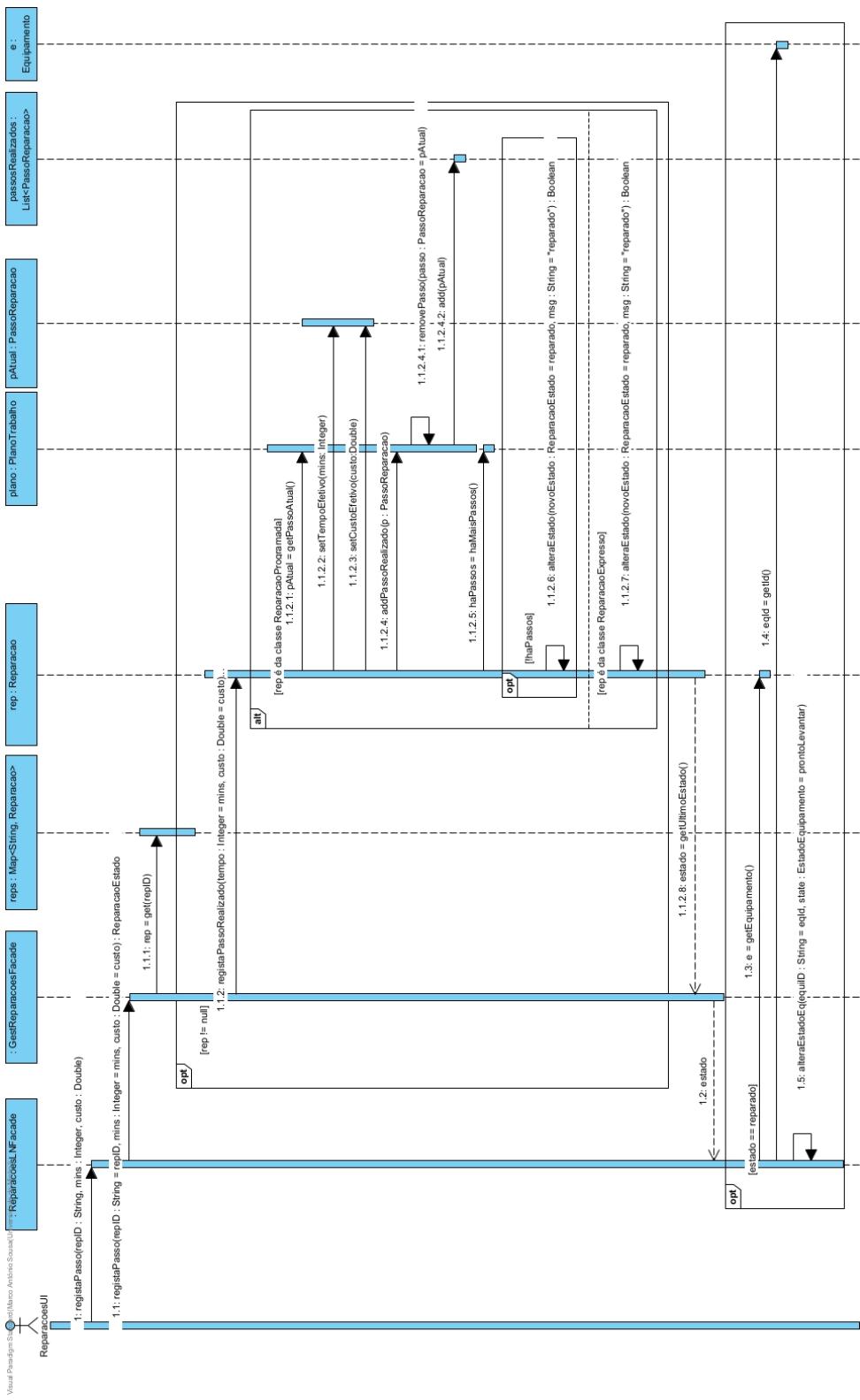


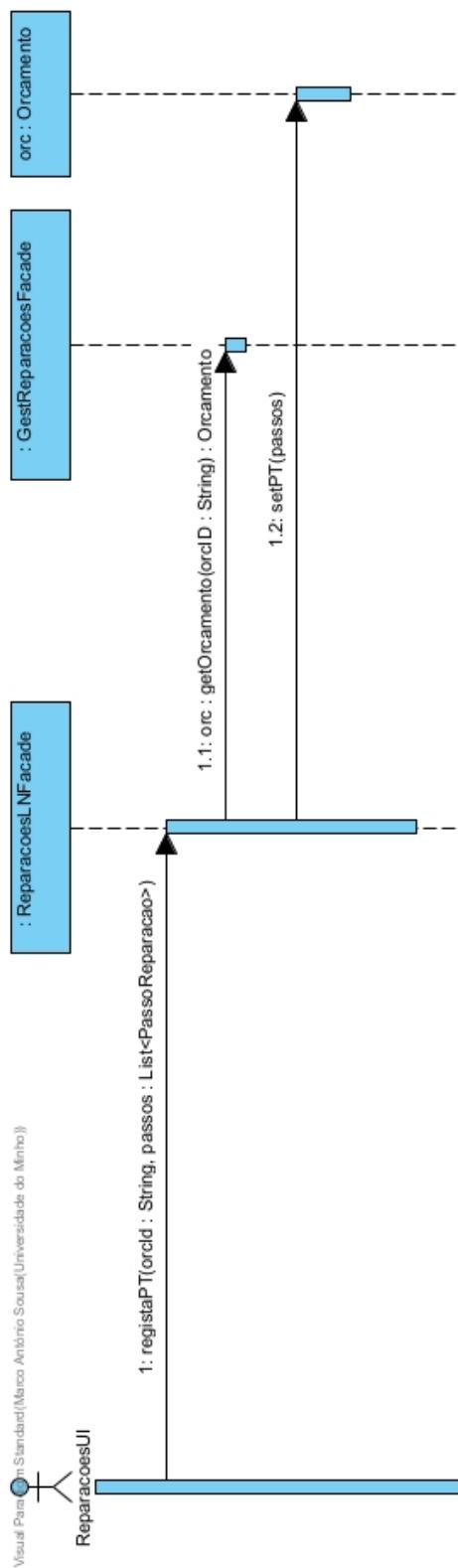


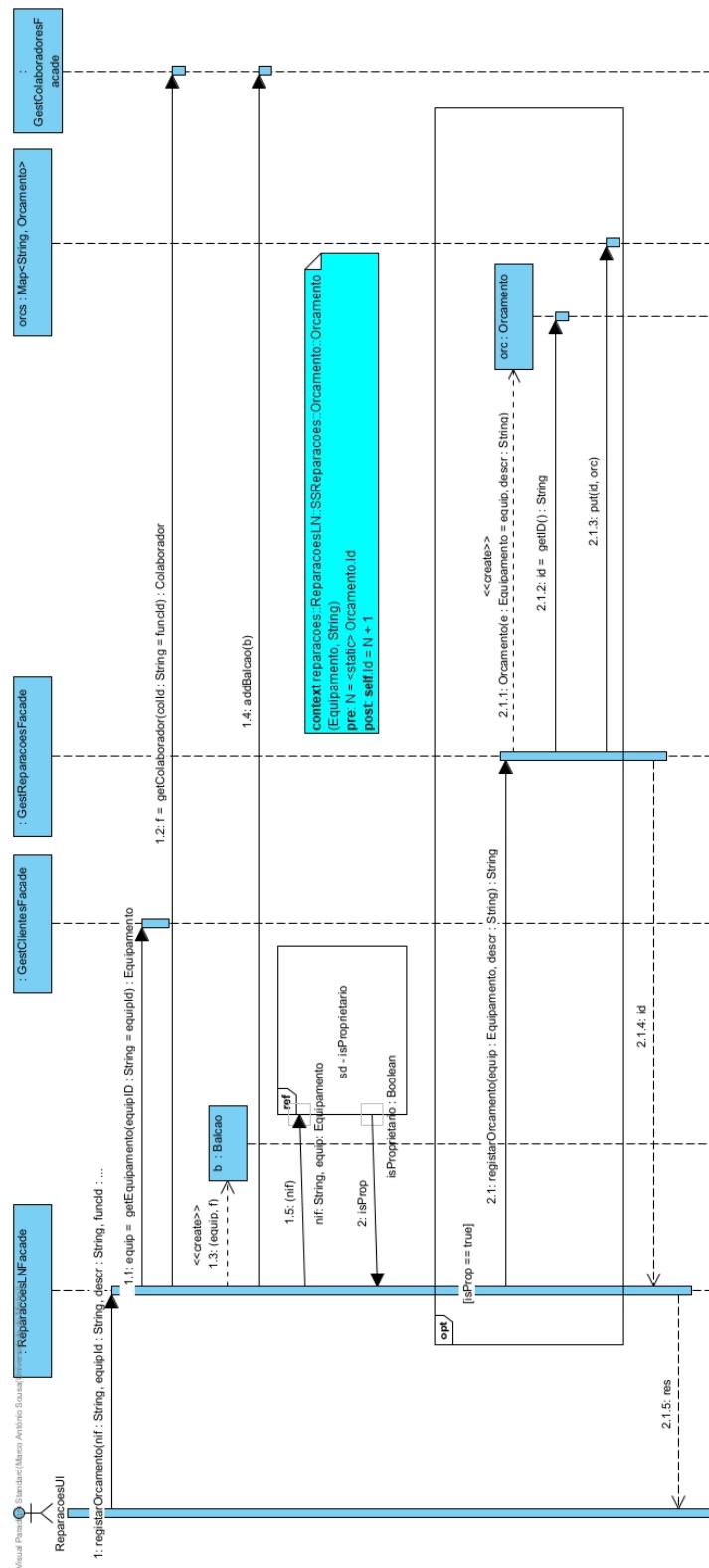


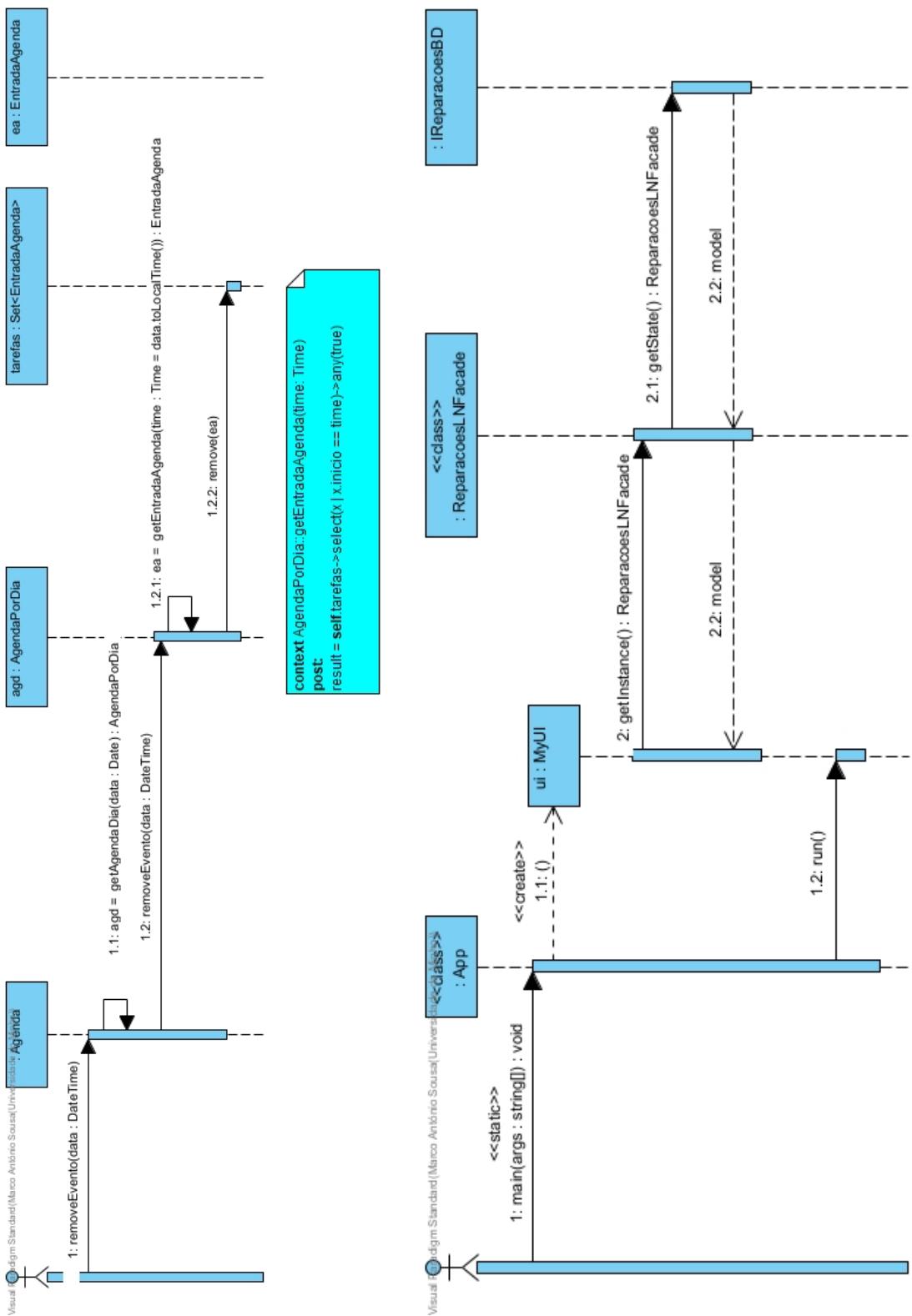


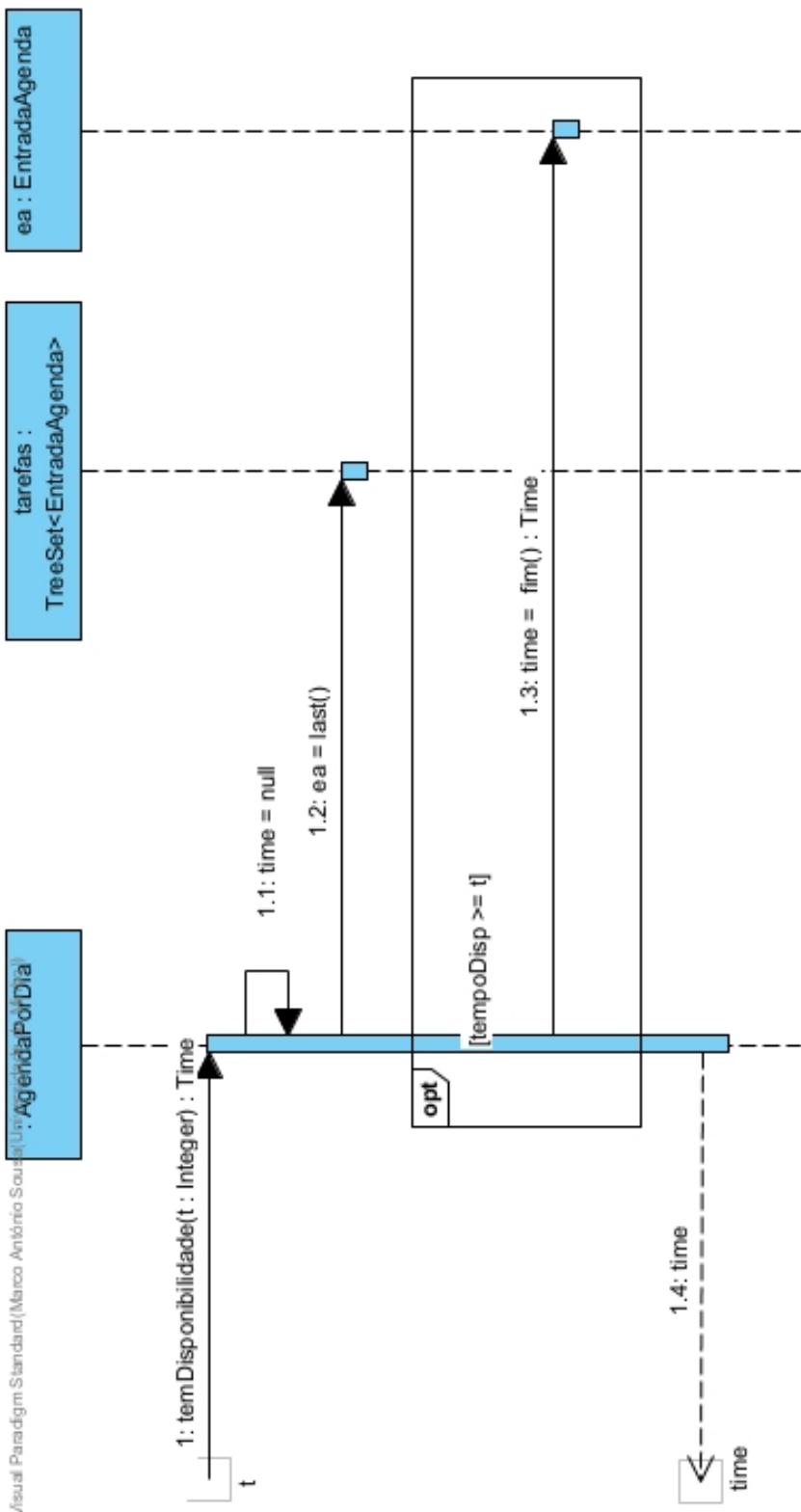


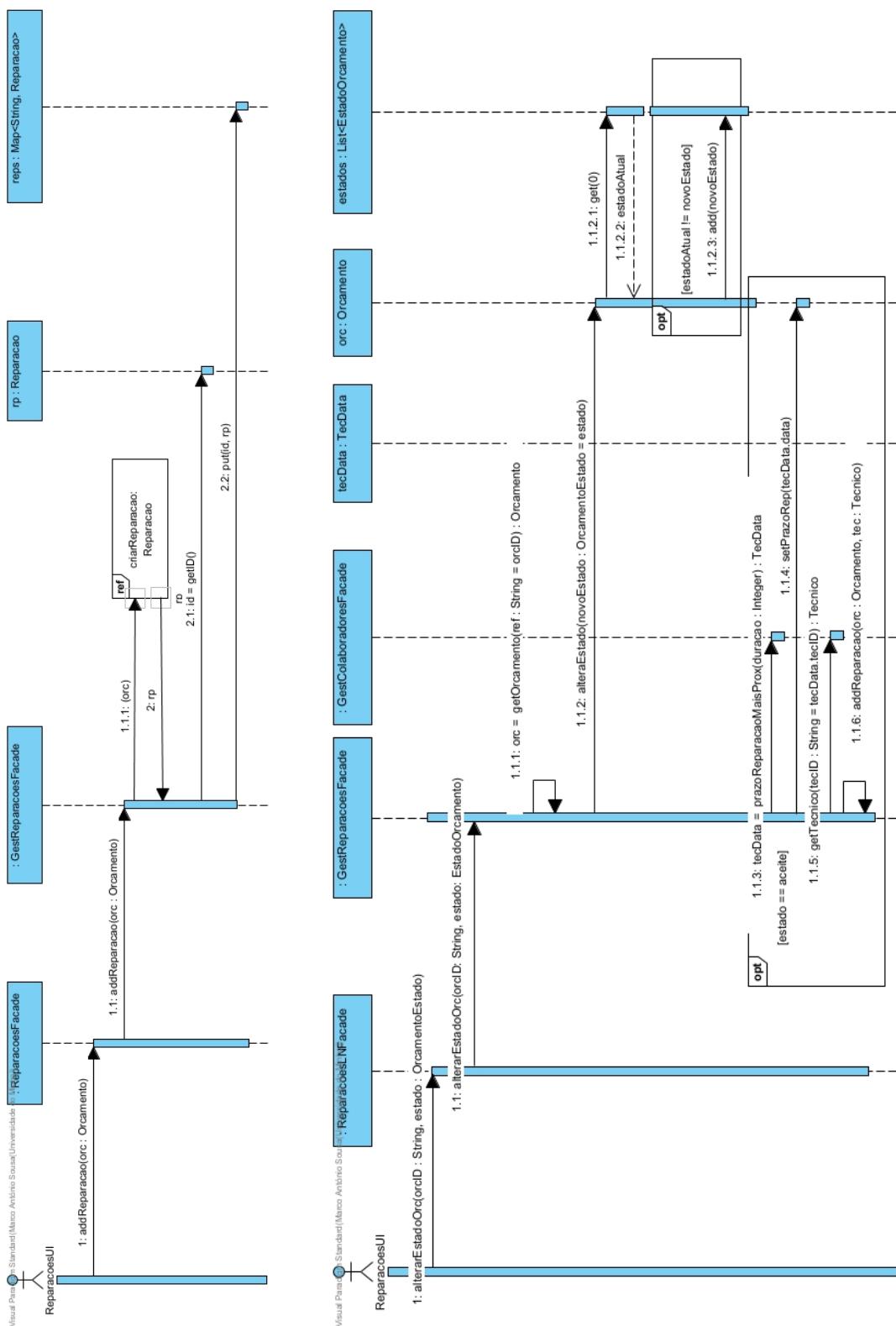


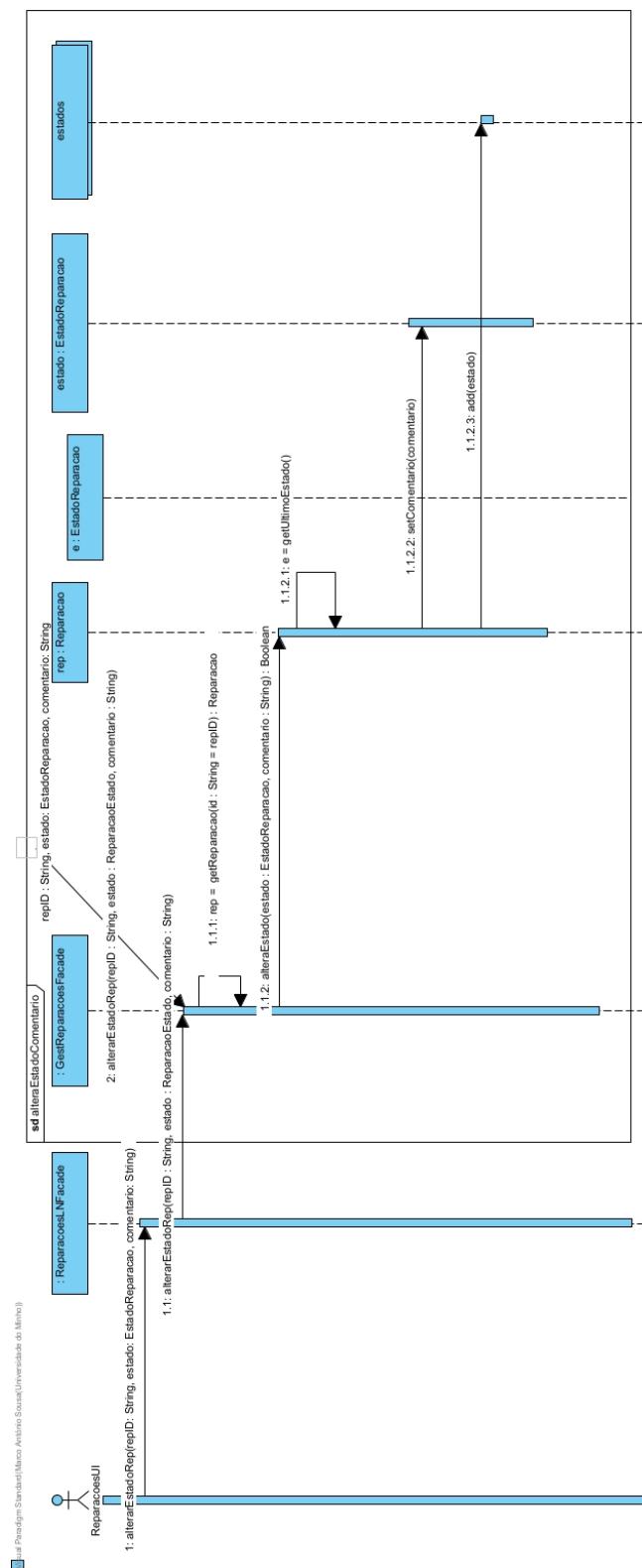


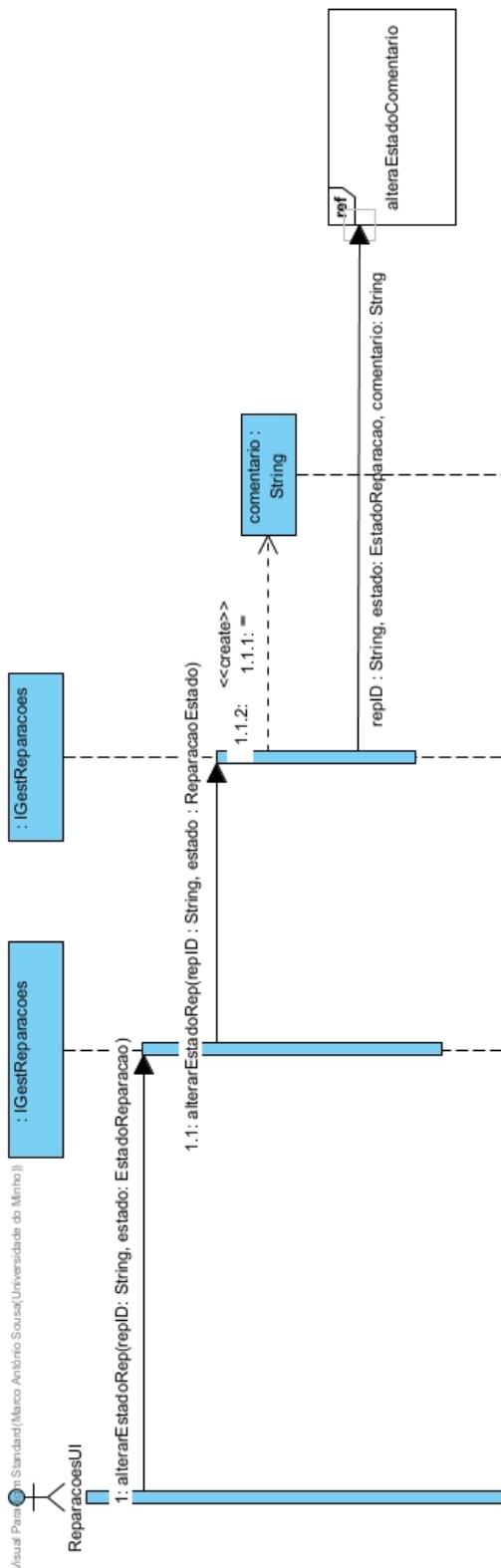


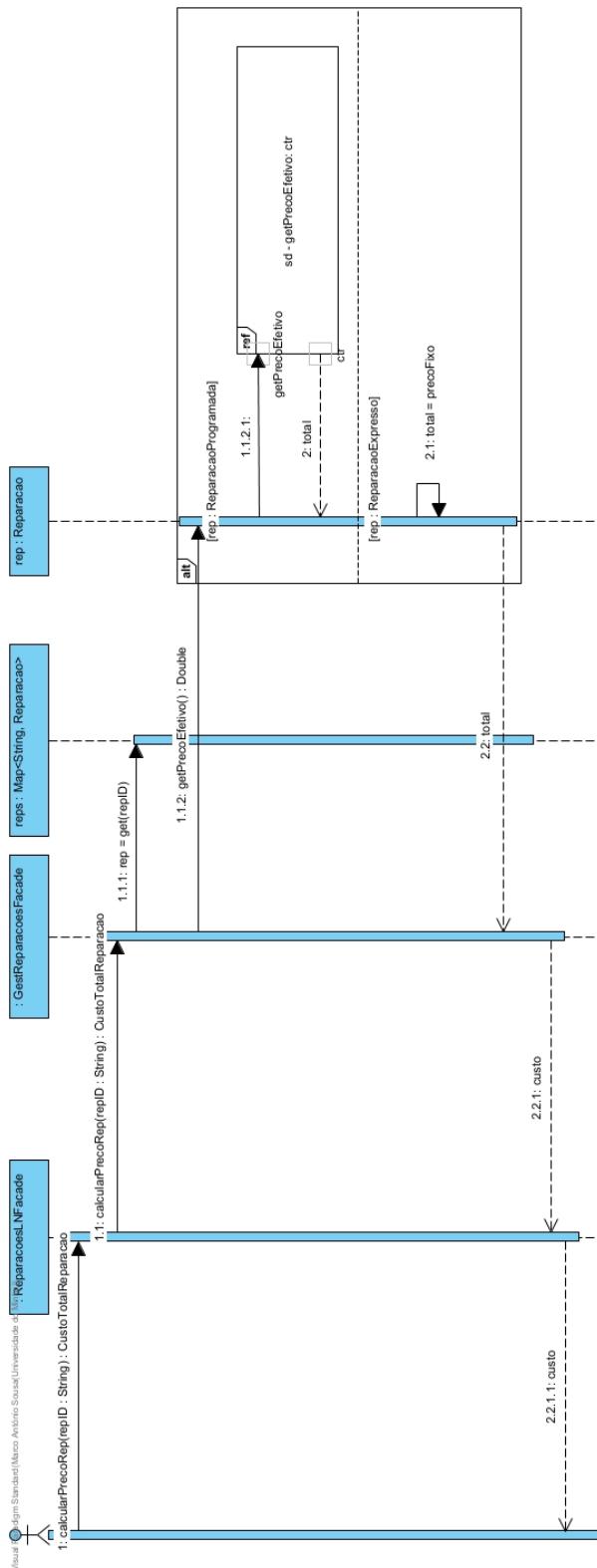


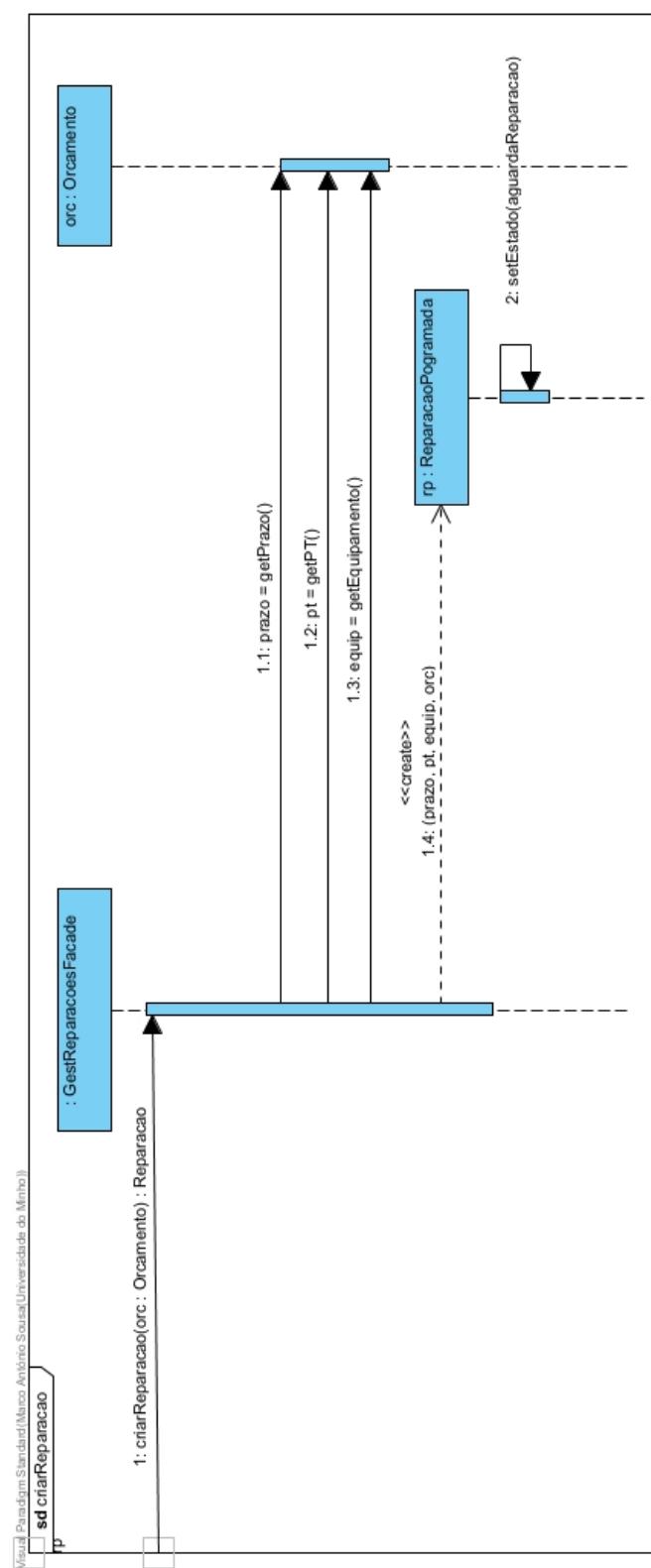


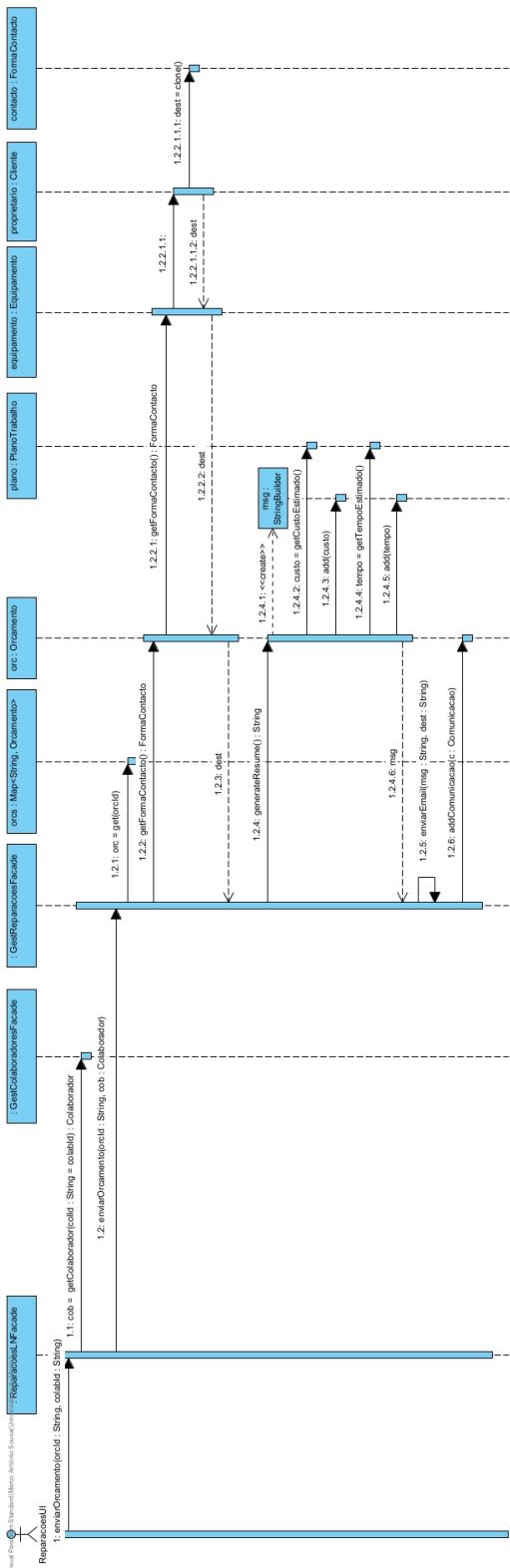


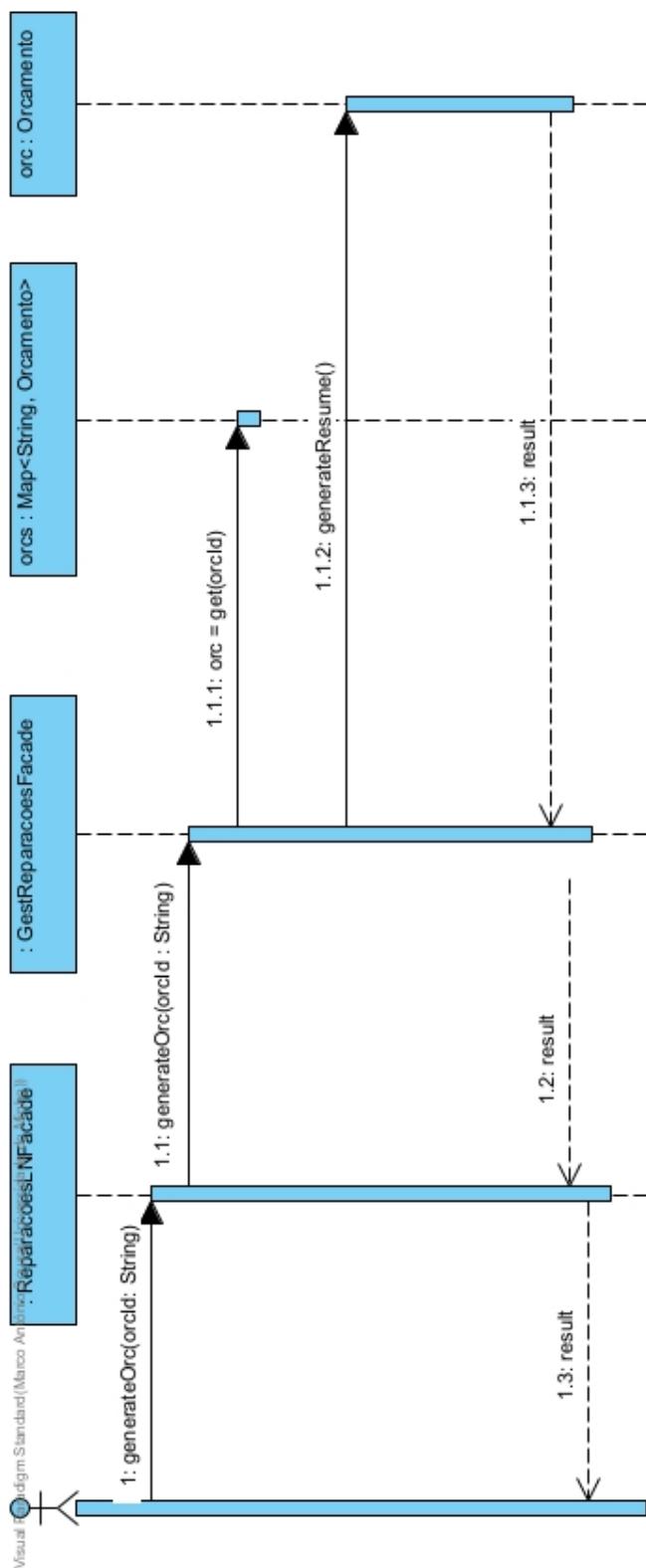


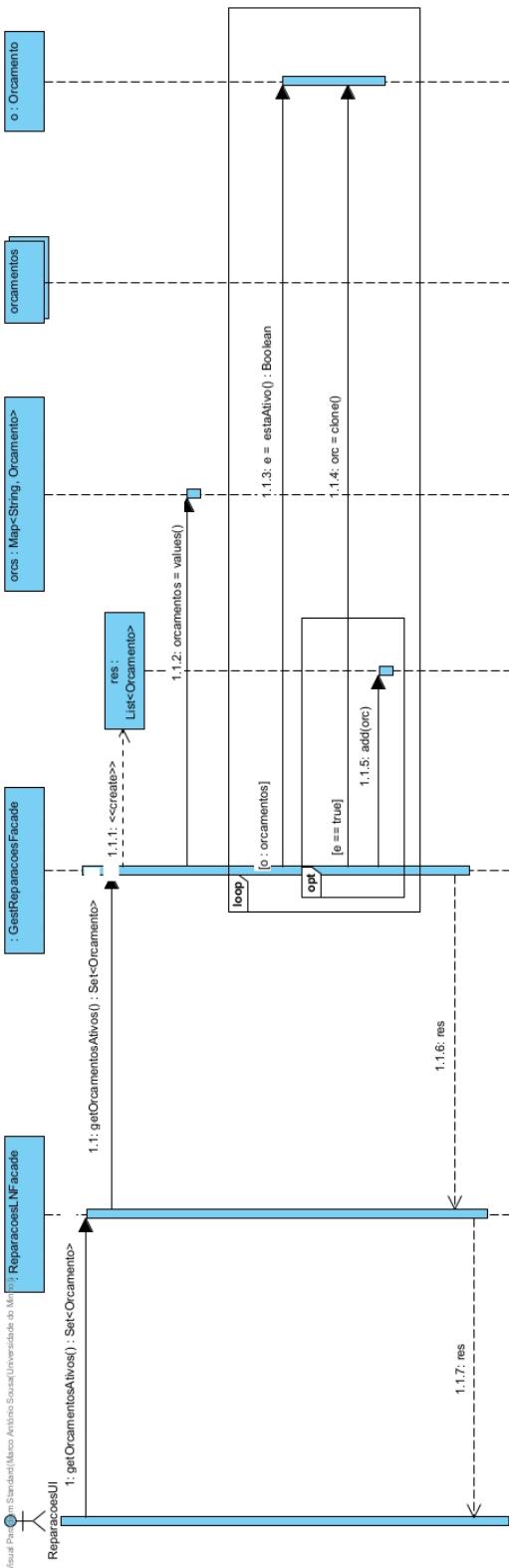












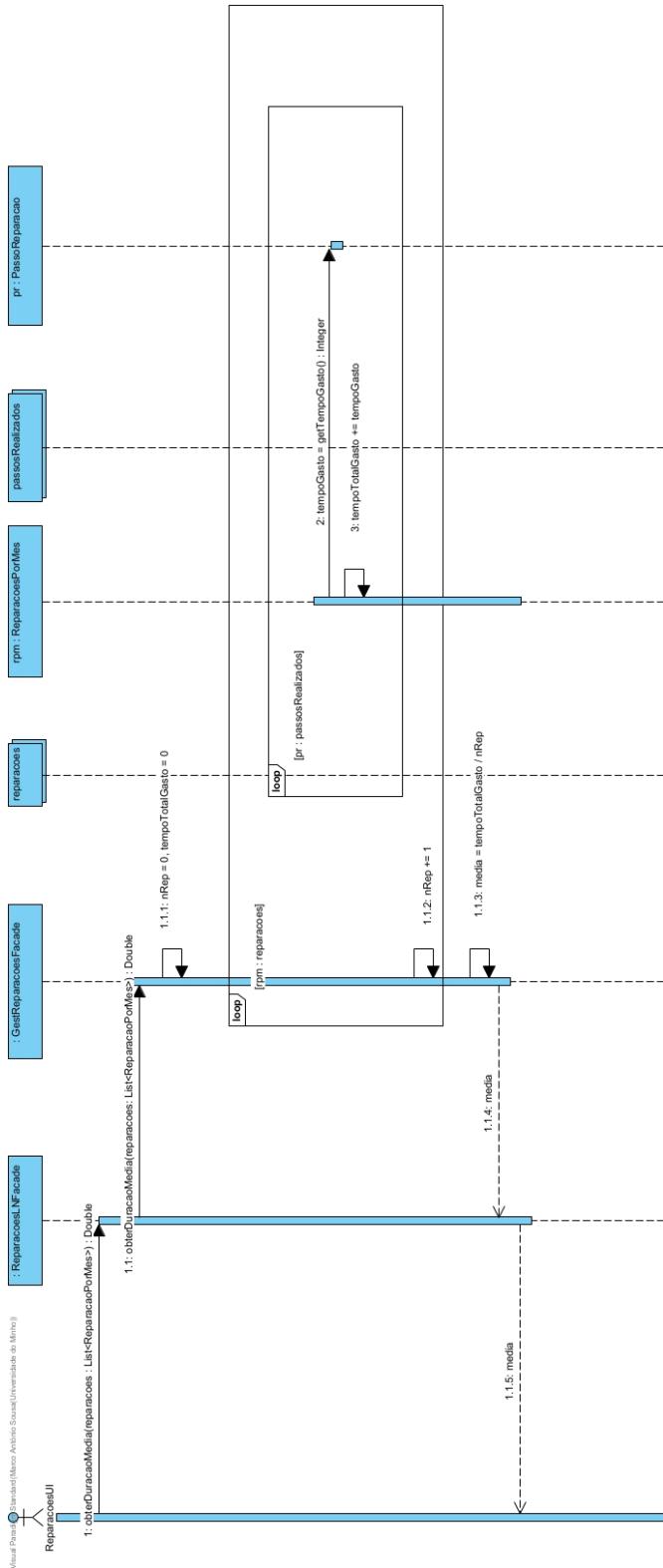


Diagrama de classe

