



IPG

Politécnico  
|da|Guarda

Polytechnic  
of Guarda

# Gestão Hospitalar

## Soft & PNet

Alunos	Bárbara Ovelheiro, nº1012526 Cristiano Patrício, nº1012390 Nelson Monteiro, nº1011304
Curso	Licenciatura em Engenharia Informática, 3ºAno
Unidades Curriculares	Engenharia de Software II Programação para a Internet
Ano Letivo	2018/2019
Docentes	Maria Clara Silveira Noel Lopes
Data	29/11/2018

## Índice

1. Descrição do tema do projeto.....	4
2. Padrões de desenvolvimento .....	5
2.1 User Value Transactions.....	5
2.2 Balanced Team .....	6
4.1 Two Tier Review .....	8
5. Atores do sistema .....	10
6. Análise de Requisitos.....	12
6.1 Diagrama de Contexto.....	12
6.2 Diagrama Casos de Uso.....	13
6.3 Descrição dos Casos de Uso.....	14
6.3.1 Descrição do caso de uso: introduzir tratamentos e equipamentos disponíveis .....	14
4.3.2 Descrição do caso de uso: consultar horário.....	15
4.3.3 Descrição do caso de uso: pedir alteração horário (turnos) .....	16
4.3.4 Descrição do caso de uso: pedir alteração horário (turnos) .....	17
4.3.5 Descrição do caso de uso: Gerar Escala Médicos .....	18
4.3.6 Descrição do caso de uso Gerar Escala Enfermeiros .....	19
4.3.7 Descrição do caso de uso autorizar pedido de trocas de turnos.....	20
4.4 Diagrama de Classes .....	21
4.5 Diagramas de Sequência.....	22
5. Algoritmo.....	30
5.1 Algoritmo gerar escalas .....	30
5.2 Algoritmo gerar horários.....	31
6. Protótipos .....	33

## Índice de Tabelas

Tabela 1 - Descrição do papel dos atores no sistema .....	10
Tabela 2 - Tabela casos de uso .....	11

## Índice de Figuras

Figura 1 - Diagrama de Contexto.....	12
Figura 2 - Diagrama casos de uso.....	13
Figura 3 - Diagrama de Classes .....	21
Figura 4 - Diagrama de Sequência Registrar Enfermeiro .....	22
Figura 5 - Diagrama de Sequência Registrar médico.....	22
Figura 6 - Diagrama de Sequência Consultar horário.....	23
Figura 7 - Diagrama de Sequência Gerar escala enfermeiros .....	24
Figura 8 - Diagrama de Sequência Introduzir regras de escalonamento de horários .....	25
Figura 9 - Diagrama de Sequencia Gerar escala médico .....	26
Figura 10 - Diagrama de Sequência Autorizar pedido de troca de turno .....	27
Figura 11 - Diagrama de Sequencia Pedir alteração turno .....	28
Figura 12 - Diagrama de Sequência Criar tratamento.....	29

# 1.Descrição do tema do projeto

O presente projeto, desenvolvido nas Unidades Curriculares: Engenharia de Software II e Programação para a Internet, do 3ºano da Licenciatura de Engenharia Informática, visa desenvolver uma aplicação web em ASP.NET Core MVC e C#, bem como outras tecnologias web como por exemplo: bootstrap, jquery, javascript, html5 e css3.

Agile é a metodologia utilizada no desenvolvimento deste projeto e por isso é realizada a análise de requisitos na Unidade Curricular Engenharia de Software II e, o desenvolvimento na Unidade Curricular Programação para a Internet.

A aplicação web tem como principais funcionalidades:

- Introduzir tratamentos e equipamentos disponíveis;
- Gerar escala e horários de médicos, enfermeiros e pacientes;
- consultar horário;
- pedir alteração de horário.

## 2. Padrões de desenvolvimento

### 2.1 User Value Transactions

#### 1. Introdução

Um conjunto de casos de uso que deve capturar os serviços importantes que os utilizadores e partes interessadas precisam num sistema. É relativamente fácil identificar transações de baixo nível, embora possa ser difícil identificar serviços úteis. Casos de uso precisam ser relativamente estáveis porque eles formam “pontos de ancoragem” para o resto o processo de desenvolvimento do produto. Os utilizadores querem ver facilmente como o sistema atenderá aos seus objetivos.

#### 2. Problema

Um sistema é deficiente se não puder fornecer serviços importantes para os seus utilizadores e não suporta as metas e objetivos especificados pela visão do sistema. Precisamos de escrever casos de uso que atendam às necessidades reais do utilizador caso desejemos vender o nosso sistema. Embora esta afirmação pareça afirmar o óbvio, existem vários fatores que levam as pessoas a escrever casos de uso que não atendem às necessidades do utilizador, como se fossem “extras”.

#### 3. Solução

Recolher o menor conjunto de objetivos que identificam os serviços mais importantes que o sistema deve entregar aos atores para satisfazer as suas intenções.

#### 4. Conclusão

Perguntar “Qual o valor que este serviço fornece ao utilizador ou partes interessadas?” Livrar dos serviços que não agregam valor ao sistema.

Não queremos desperdiçar tempo valioso a escrever casos de uso ou a implementar código para um recurso que ninguém irá usar ou não tenha interesse em.

## 2.2 Balanced Team

### 1. Introdução

BalancedTeam é um padrão de desenvolvimento direcionado para a equipa. A imagem acima representada retrata bem o conceito do padrão BalancedTeam. Uma equipa constituída por pessoas com diferentes formações técnicas tem a vantagem de ter uma visão, cultura e perspetivas diferenciadoras no mercado. Nas balanced teams cria-se uma sinergia entre os membros que pode ser uma mais valia na resolução de problemas, existindo a possibilidade de chegar a várias soluções para o mesmo problema. Nos próximos tópicos abordarei o problema e a solução para o mesmo com o padrão de desenvolvimento BalancedTeam.

### 2. O Problema

Um dos principais fatores que leva ao fracasso de algumas startups é a existência de unbalanced teams, ou equipas “desequilibradas”. Uma equipa constituída por pessoas com mentalidade e personalidade semelhantes na forma de desenvolver casos de uso (use cases), muitas vezes não consegue avaliar as necessidades de outros stakeholders, o que pode resultar num conjunto de casos de uso limitados e restritos, que não satisfazem as necessidades.

Escrever casos de uso significativos requer criatividade de várias pessoas com diferentes pontos de vista. Indivíduos com a mesma formação técnica em comum tendem a focar-se essencialmente nos mesmos tipos de problemas, o que faz com que usem a mesma técnica para os resolver. Embora a abordagem ao problema possa fornecer uma solução satisfatória, é esperado, por parte dos membros da equipa, um pensamento out-of-the-box, que é considerado uma skill fundamental nesta fase de desenvolvimento do produto. Cada profissão usa um vocabulário específico que outras pessoas que não são da área não entendem. Se na escrita dos casos de uso é usado vocabulário excessivamente técnico, pode resultar em documentos confusos para pessoas com outras formações técnicas. Mesmo que o cliente entenda a generalidade de alguns conceitos, não consegue compreender as suas implicações e pode simplesmente ignorá-los ou interpretar mal os pontos-chave. As equipas precisam de diferentes conhecimentos em diversos momentos do desenvolvimento do produto. Se houver um especialista no assunto no início de um novo projeto, tanto melhor. As boas equipas são balanceadas para que as especialidades de um membro colmatem alguma fraqueza por parte de um outro membro. Uma equipa constituída por membros com skills complementares é melhor para definir casos de uso que sejam suportados por vários tipos de utilizadores. O

desenvolvimento de software melhora à medida que melhoramos as skills pessoais e melhoramos a eficácia de colaboração da equipa.

### 3. A solução

Constituir uma equipa com pessoas de diferentes especialidades para conseguir alcançar com sucesso os interesses dos stakeholders no processo de desenvolvimento. Garantir que a equipa seja constituída por programadores e utilizadores finais. A tendência é sempre usar os melhores programadores e engenheiros de software para escrever todos os casos de uso. Devemos criar uma equipa onde vamos articular pessoas com diferentes especialidades e pontos de vista. Se o produto que estivermos a desenvolver implicar questões de natureza jurídica, é aconselhável incorporar alguém da área. É importante incluir na equipa pessoas que pensem em pequenos detalhes, detalhes de baixo nível. A equipa deve ser diversificada o suficiente para fornecer pontos de vista técnicos e corporativos. É fundamental incluir pessoas que sejam boas em manter o foco e que saibam quando devem sair.

### 4. Conclusão

O maior benefício em usar balanced teams passa por envolver pessoas com diferentes backgrounds e perspetivas, que em cooperação conseguem escrever casos de uso usando uma terminologia comum e compreensível. Embora importantes, as balanced teams constituem apenas uma pequena percentagem daquele que será o “quadro”. Uma equipa necessita também de seguir um bom processo de desenvolvimento para escrever casos de uso de qualidade.

## 4.1 Two Tier Review

### Two Tier Review

O método de avaliação de casos de uso Two Tier Review assemelha-se ao sistema democrático, onde um determinado número de pessoas representa um grupo permitindo assim que haja discussão sobre as preocupações de todos, mas também eficiência. As revisões são um fator importante para manter o rigor e avaliar o progresso da tarefa em mãos. A revisão por parte de outro que não o autor é por norma mais eficiente na procura de erros.

Os casos de uso são um papel fundamental na organização do projeto e como tal a consulta com o cliente relativamente aos mesmos deve ser feita logo desde início de forma a evitar repetição de trabalho e manter o foco relativamente às preocupações do cliente.

É também incorporado o método "Small Writing Team" de forma a evitar ineficiências e falta de coordenação, no entanto deverá haver uma audiência capaz de proporcionar experiência e visão dos interesses das partes interessadas.

Apesar das revisões serem importantes para o avanço e manutenção do projeto estas podem ser caras e consumidoras de tempo devendo ser apenas realizadas quando estritamente necessárias e com a menor quantidade de pessoas possível.

Quando o sistema é demasiado grande ou complexo devem ser feitas revisões internas através de grupos de revisão dispersos para diferentes funções. Após as revisões internas serem realizadas é feita uma avaliação geral por um grupo externo.

A primeira camada de revisão deve avaliar o funcionamento do sistema interno de forma a que a segunda camada possa focar-se em juntar as funcionalidades.

Este grupo varia para cada projeto e tem como principal objetivo averiguar o seguinte:

- Valor comercial
- Validade das especificações
- Realista para implementação

Exemplos:



Wings Over the World: A excessiva quantidade de revisões envolvendo os clientes pode-se tornar cansativa para os mesmos. Estes pretendem manter a sua visão no projeto, mas, no entanto, este excesso é desnecessário para pequenas alterações.

The Programmer Who Cried Review:

Longas horas de revisões são pouco ineficientes e propensas a erros, deverá haver uma melhor organização e planeamento

## 5. Atores do sistema

Atores	Descrição do papel do ator no sistema
Paciente	O paciente não interage diretamente com o sistema
Médico	O médico é um utilizador final do sistema que pode: introduzir os tratamentos e a capacidade disponível de equipamentos; consultar o horário; pedir alteração do horário dos turnos; e receber horário semanal.
Enfermeiro	O enfermeiro é um utilizador final do sistema que pode: Consultar escala de tratamentos do paciente; consultar horário; pedir alteração do horário dos turnos; e receber horário semanal.
Diretor de Serviço	O diretor de serviço é um utilizador final do sistema que pode: introduzir regras de escalonamento de horários; gerar a escala dos pacientes, médicos e enfermeiros; e autorizar pedido de troca de turno
Máquina de Ponto	A máquina de ponto interage diretamente com o sistema registando a hora de entrada e saída de um médico e de um enfermeiro.

Tabela 1 - Descrição do papel dos atores no sistema

Atores	Objetivos
Paciente	Receber escala de tratamentos
Médico	Introduzir os tratamentos e capacidade disponível(equipamentos)
	Consultar horário
	Pedir alteração no horário (Turnos)
	Receber horário semanal
Enfermeiro	Consultar escala de tratamentos do paciente
	Consultar horário
	Pedir alterações no horário (Turnos)
	Receber horário semanal
Diretor de Serviço	Introduzir regras de escalonamento de horários
	Gerar escala de pacientes, médicos e enfermeiros
	Autorizar pedido de troca de turnos
Máquina de Ponto	Registrar hora de entrada de um enfermeiro e de um médico
	Registrar hora de saída de um enfermeiro e de um médico

Tabela 2 - Tabela casos de uso

## 6. Análise de Requisitos

### 6.1 Diagrama de Contexto

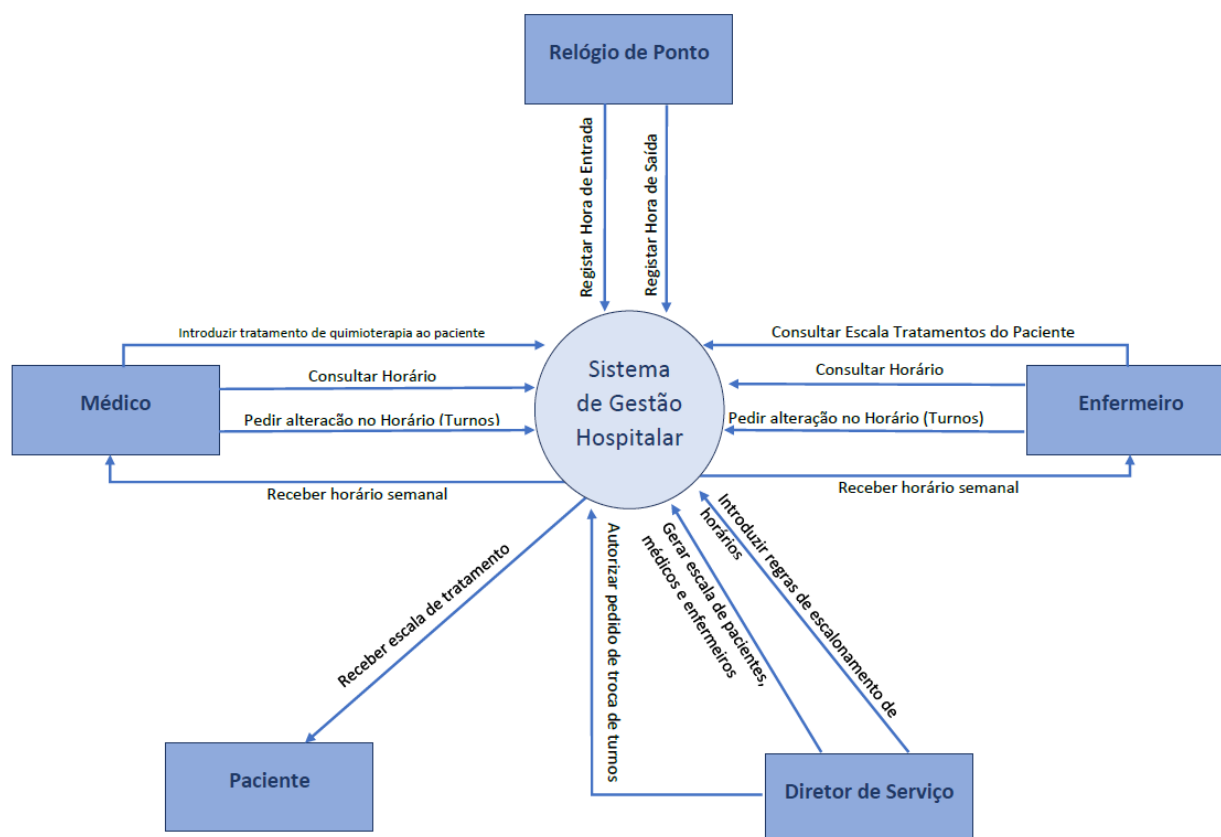


Figura 1 - Diagrama de Contexto

## 6.2 Diagrama Casos de Uso

### Escalonamento Hospitalar

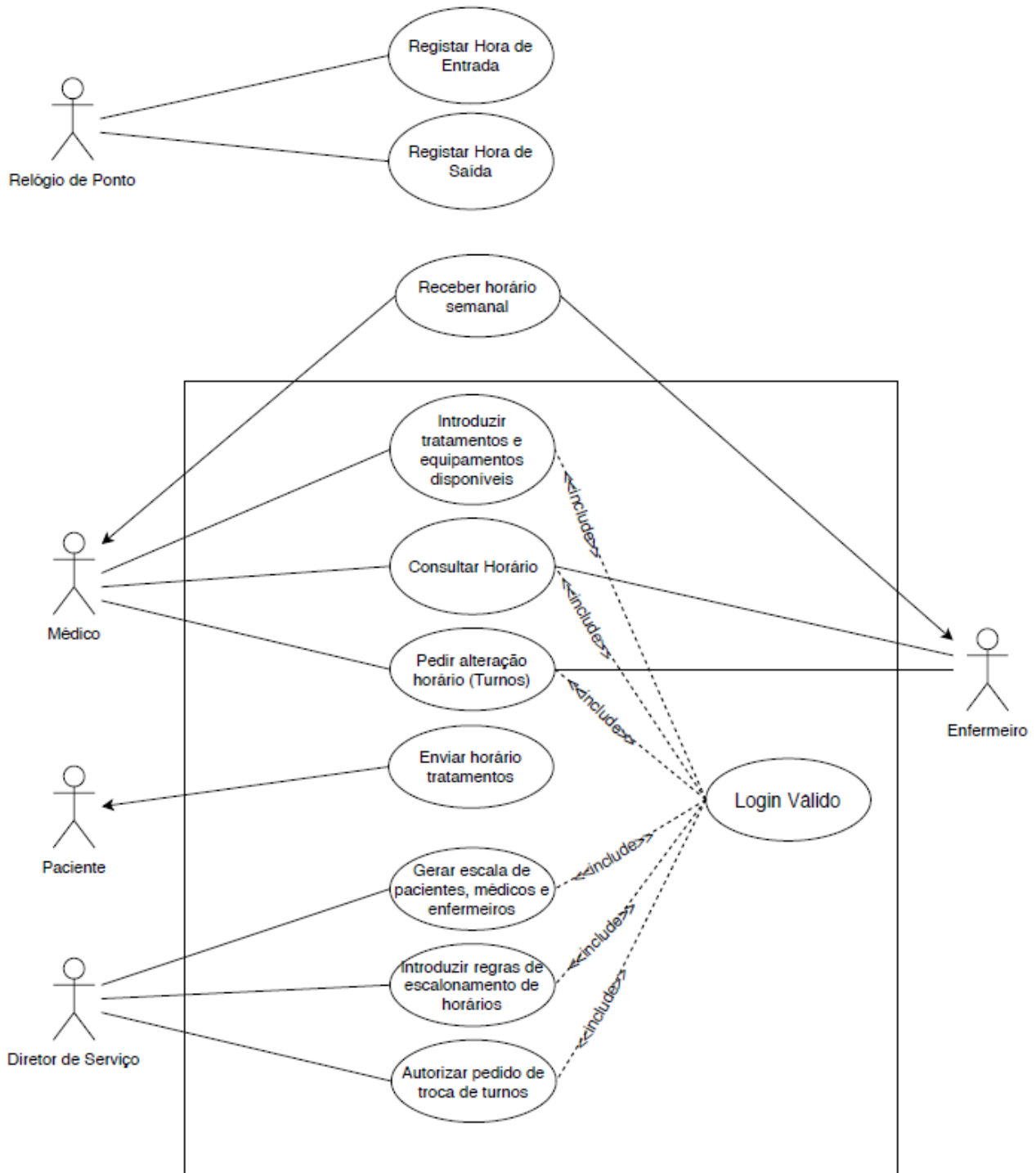


Figura 2 - Diagrama casos de uso

## 6.3 Descrição dos Casos de Uso

### 6.3.1 Descrição do caso de uso: introduzir tratamentos e equipamentos disponíveis

**Nome:** Introduzir tratamentos e equipamentos disponíveis.

**Descrição:** O médico introduz no sistema um determinado tratamento destinado a um paciente, bem como o número de equipamentos necessários para o respetivo tratamento.

**Tamanho:** L

**Pré-condição:** O médico tem de ter um login válido.

**Caminho principal:**

1. O médico seleciona a opção “Inserir um novo tratamento”;
2. O médico pesquisa pelo número de utente de saúde;
3. O médico insere a patologia;
4. O médico seleciona o grau da patologia;
5. O médico insere a duração do ciclo;
6. O médico seleciona o regime
7. O médico insere a data de início do ciclo;
8. O médico insere a data de fim do ciclo;
9. O médico insere o medico que vai acompanhar o tratamento.

**Caminho alternativo:**

1. A pesquisa do número de utente de saúde é um campo obrigatório;
2. A patologia é um campo obrigatório;
3. O grau da patologia é um campo obrigatório;
4. A data de início é um campo obrigatório;
5. A data de fim é um campo obrigatório;
6. A duração do ciclo é um campo obrigatório;
7. O regime é um campo obrigatório;
8. O sistema cancela quando o utilizador não insere o novo ciclo de tratamentos.

**Suplementos ou adornos:**

- **Teste 1:** Validar o campo Patologia: se o utilizador inserir no campo da Patologia a seguinte expressão: lesão998qletal; o sistema notifica o utilizador que não é uma patologia válida.

- **Teste 2:** Validar o campo Duração do ciclo: se o utilizador deixar em branco este campo, o sistema notifica o utilizador que tem de introduzir a duração do ciclo, isto é, não pode deixar o campo em branco.

**Pós Condições:** o sistema guarda na base de dados o novo ciclo de tratamentos do paciente.

### 4.3.2 Descrição do caso de uso: Consultar horário

**Nome:** Consultar Horário.

**Descrição:** O enfermeiro faz a consulta do seu horário semanal ou mensal.

**Tamanho:** L

**Pré-condição:** O enfermeiro tem de ter um login válido.

**Caminho principal:**

1. O enfermeiro seleciona a opção “Consultar horário”;
2. O sistema apresenta o formulário.
3. O enfermeiro seleciona a data em que pretende visualizar o horário;
4. O enfermeiro seleciona a opção “Consultar”;
5. O sistema faz a pesquisa na Base de Dados e devolve o horário.

**Caminho alternativo:**

- 4.1 O sistema apresenta a mensagem de erro: “**Não existem horários disponíveis para visualização no intervalo de datas selecionadas**”;

**Suplementos ou adornos:**

- Validação da data;

**Pós Condições:** O sistema apresenta o horário ao utilizador.

### 4.3.3 Descrição do caso de uso: Pedir alteração horário (turnos)

**Nome:** Pedir alteração horário (turnos).

**Descrição:** O médico introduz um pedido de alteração de horário (turno).

**Tamanho:** L

**Pré-condição:** O médico tem de ter um login válido.

**Caminho principal:**

1. O médico seleciona a opção “Pedir alteração turno”;
2. O sistema apresenta o formulário;
3. O médico escolhe a data do turno a trocar;
4. O sistema devolve os horários para a data selecionada;
5. O médico escolhe o turno a trocar;
6. O médico escolhe a data do turno para troca.
7. O sistema devolve os horários para a data selecionada.
8. O sistema insere o registo na tabela “PedidoTrocaTurno”.
9. O sistema devolve mensagem de sucesso.

**Caminho alternativo:**

- 4.1. O sistema apresentada a mensagem “**Horários não encontrados para a data selecionada**”;
- 7.1. O sistema apresentada a mensagem “**Horários não encontrados para a data selecionada**”;

**Suplementos ou adornos:**

Validação da data;

**Pós Condições:**

O sistema insere na tabela “PedidoTrocaTurno” o registo.



#### 4.3.4 Descrição do caso de uso: Pedir alteração horário (turnos)

**Nome:** Pedir alteração horário (turnos).

**Descrição:** O enfermeiro introduz um pedido de alteração de horário (turno).

**Tamanho:** L

**Pré-condição:** O enfermeiro tem de ter um login válido.

**Caminho principal:**

10. O enfermeiro seleciona a opção “Pedir alteração turno”;
11. O sistema apresenta o formulário;
12. O enfermeiro escolhe a data do turno a trocar;
13. O sistema devolve os horários para a data selecionada;
14. O enfermeiro escolhe o turno a trocar;
15. O enfermeiro escolhe a data do turno para troca.
16. O sistema devolve os horários para a data selecionada.
17. O sistema insere o registo na tabela “PedidoTrocaTurno”.
18. O sistema devolve mensagem de sucesso.

**Caminho alternativo:**

- 4.2. O sistema apresentada a mensagem “**Horários não encontrados para a data selecionada**”;
- 7.1. O sistema apresentada a mensagem “**Horários não encontrados para a data selecionada**”;

**Suplementos ou adornos:**

Validação da data;

**Pós Condições:**

O sistema insere na tabela “PedidoTrocaTurno” o registo.

### 4.3.5 Descrição do caso de uso: Gerar Escala Médicos

**Nome:** Gerar Escala Médicos

**Descrição:** O diretor de serviço introduz no sistema o número de médicos para o turno 1, turno 2 e turno 3; introduz também a data de início da semana.

**Tamanho:** XL

**Pré-condição:** O diretor de serviço tem de ter um login válido.

**Caminho principal:**

1. O diretor de serviço seleciona a opção “GerarEscalaMedico”;
2. O diretor de serviço insere o número de médicos para o turno 1;
3. O diretor de serviço insere o número de médicos para o turno 2;
4. O diretor de serviço insere o número de médicos para o turno 3;
5. O diretor de serviço insere a data de início da semana;
6. O diretor de serviço clica no botão “Gerar Escala”;
7. O sistema gera a escala do médico automaticamente.

**Caminho alternativo:**

1. Os campos dos turnos são obrigatórios;
2. O campo da data é obrigatório.

**Suplementos ou adornos:**

- **Teste 1:** Introduzir a data inferior à data atual: o sistema notifica o utilizador que não é uma data válida;
- **Teste 2:** Validar o campo turno 1, turno 2 e turno 3: o sistema notifica o utilizador não pode deixar os campos em branco.

**Pós Condições:** O sistema gera automaticamente e apresenta a escala do médico ao utilizador e aos respetivos médicos.

### 4.3.6 Descrição do caso de uso Gerar Escala Enfermeiros

**Nome:** Gerar Escala Enfermeiros

**Descrição:** O diretor de serviço introduz no sistema o número de enfermeiros para o turno 1, turno 2 e turno 3; introduz também a data de início da semana.

**Tamanho:** XL

**Pré-condição:** O diretor de serviço tem de ter um login válido.

**Caminho principal:**

1. O diretor de serviço seleciona a opção “GerarEscalaEnfermeiro”;
2. O diretor de serviço insere o número de enfermeiros para o turno 1;
3. O diretor de serviço insere o número de enfermeiros para o turno 2;
4. O diretor de serviço insere o número de enfermeiros para o turno 3;
5. O diretor de serviço insere a data de início da semana;
6. O diretor de serviço clica no botão “Gerar Escala”;
7. O sistema gera a escala do enfermeiro automaticamente.

**Caminho alternativo:**

1. Os campos dos turnos são obrigatórios;
2. O campo da data é obrigatório.

**Suplementos ou adornos:**

- **Teste 1:** Introduzir a data inferior à data atual: o sistema notifica o utilizador que não é uma data válida;
- **Teste 2:** Validar o campo turno 1, turno 2 e turno 3: o sistema notifica o utilizador não pode deixar os campos em branco.

**Pós Condições:** O sistema gera automaticamente e apresenta a escala do enfermeiro ao utilizador e aos respetivos enfermeiros.

### 4.3.7 Descrição do caso de uso autorizar pedido de trocas de turnos

**Nome:** Autorizar pedido de trocas de turnos.

**Descrição:** O diretor de serviço autoriza o pedido de troca de turno por parte de um enfermeiro ou médico.

**Tamanho:** M

**Pré-condição:** O diretor de serviço tem de ter um login válido.

**Caminho principal:**

1. O diretor de serviço seleciona a opção “Ver pedidos de troca de turnos pendentes”;
2. O sistema apresenta, caso existam, os pedidos de troca de turnos pendentes;
3. O diretor de serviço seleciona a opção “Validar troca”;
4. O sistema pede confirmação do pedido de troca através do login;
5. O diretor de serviço preenche os campos do login;
6. O sistema valida o login e apresenta a mensagem “Troca aprovada com sucesso”.

**Caminho alternativo:**

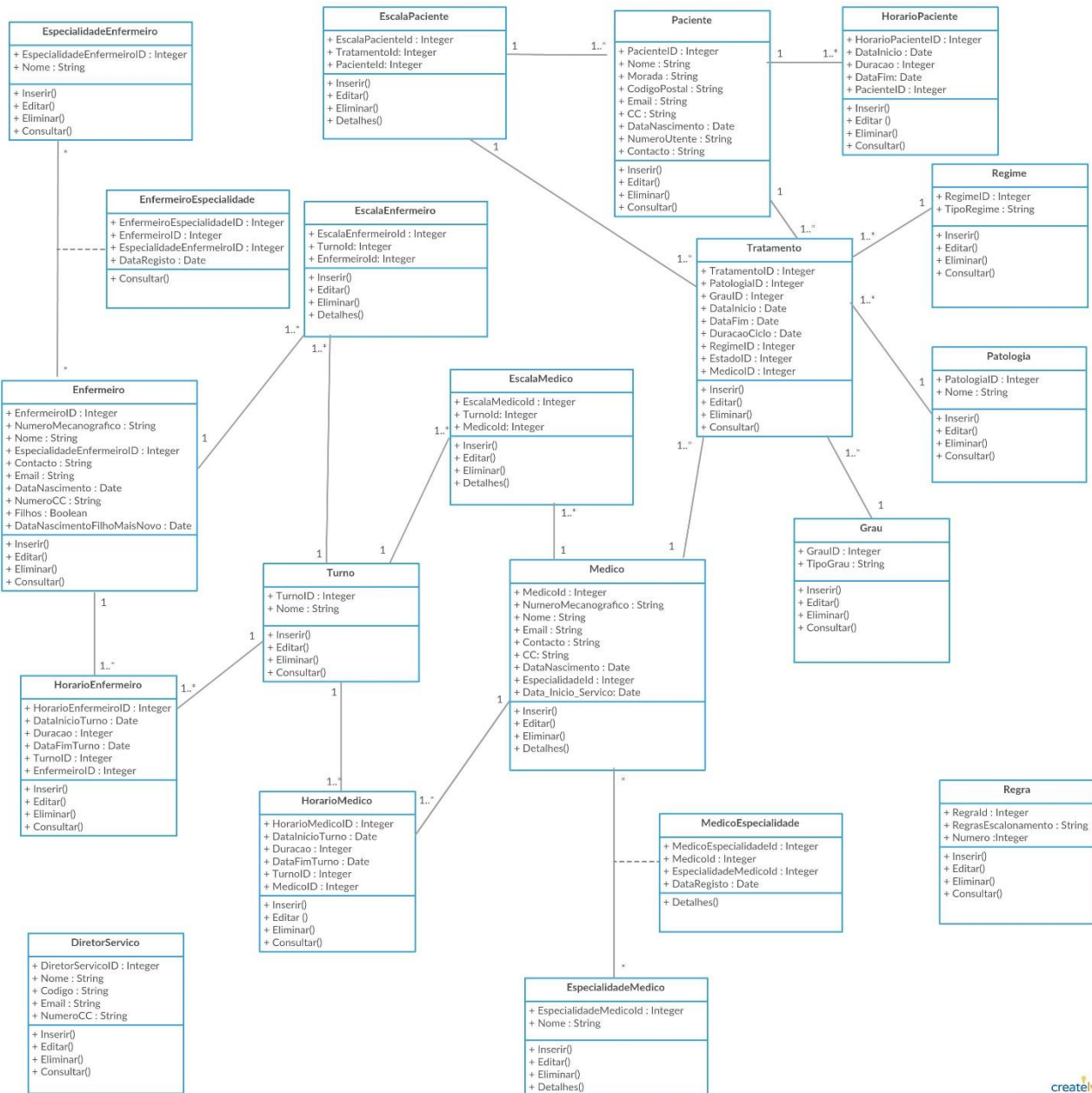
1. O sistema apresenta a mensagem “Não existem pedidos de troca de turnos pendentes”;
2. O sistema apresenta a mensagem “Login inválido. Código ou password inválidos”.

**Suplementos ou adornos:**

- Teste 1: Validar se existem registos na base de dados com os pedidos de troca de turnos;
- Teste 2: Validar login para confirmar pedido de troca;

**Pós Condições:** Alteração na tabela `HorarioEnfermeiro` ou `HorarioMedico` dos registos correspondentes ao pedido de troca de turnos.

### 4.3 Diagrama de Classes



*Figura 3 - Diagrama de Classes*

## 4.4 Diagramas de Sequência

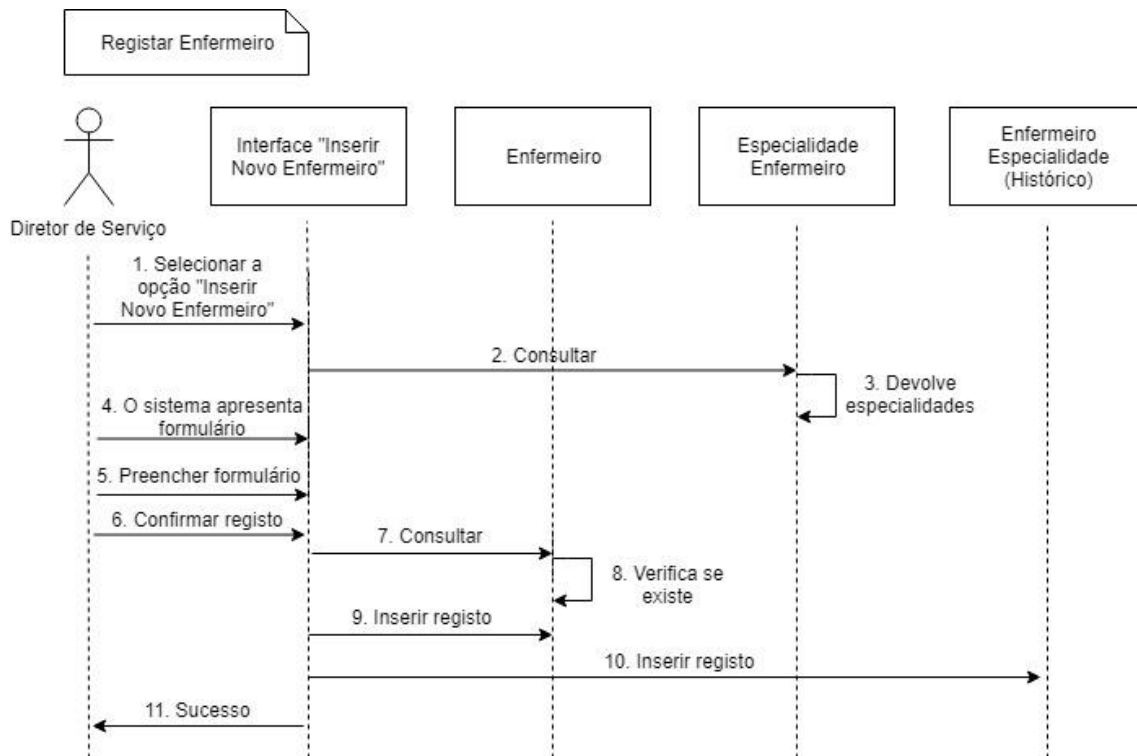


Figura 4 - Diagrama de Sequência "Registrar Enfermeiro"

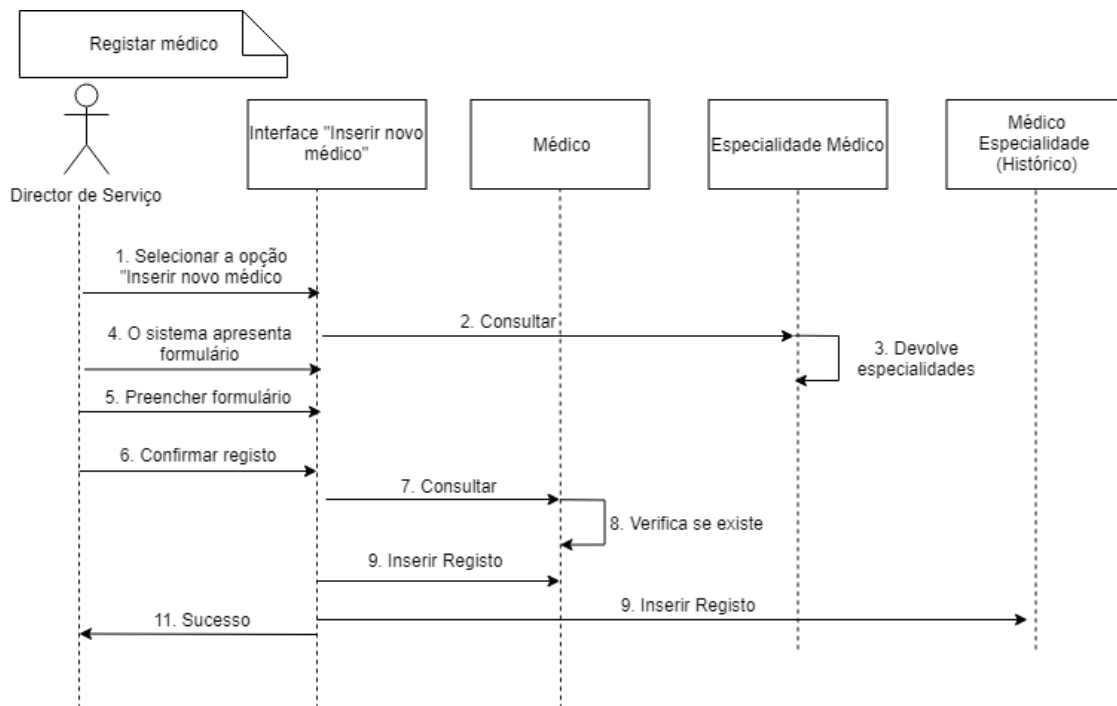


Figura 5 - Diagrama de Sequência Registrar médico

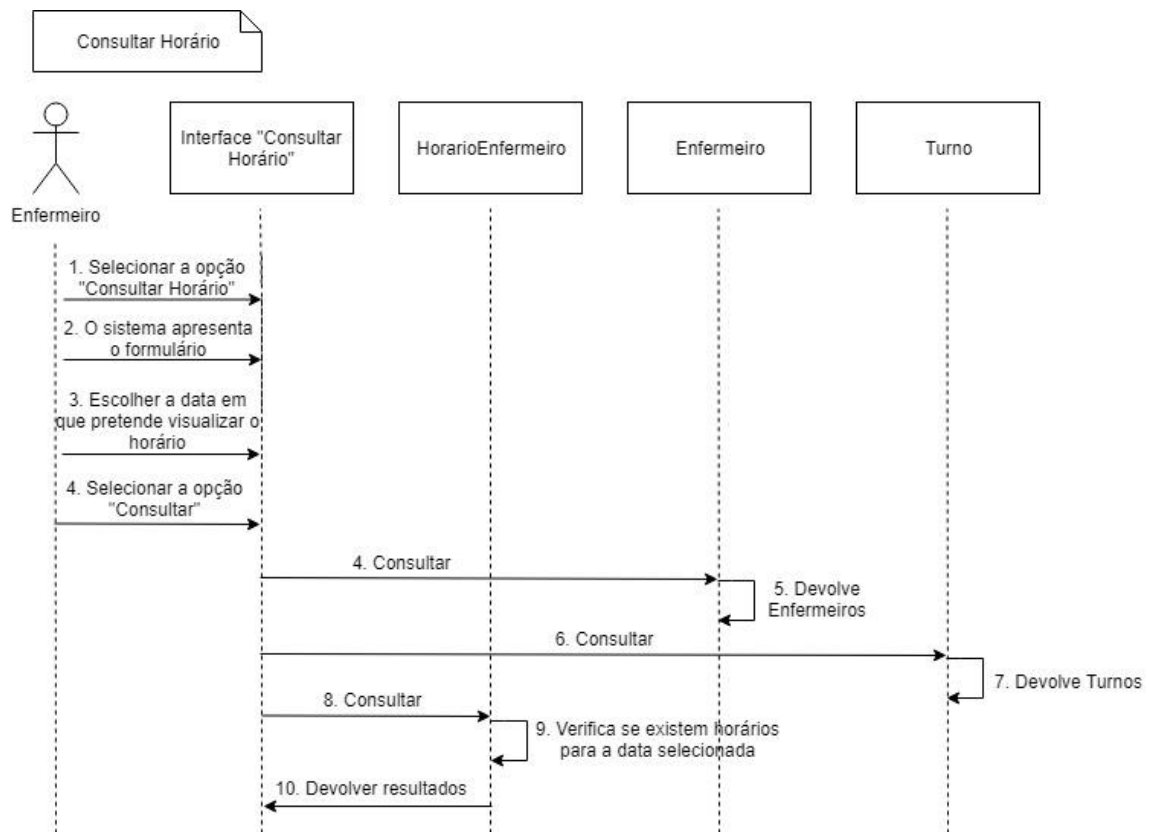


Figura 6 - Diagrama de Sequência Consultar horário

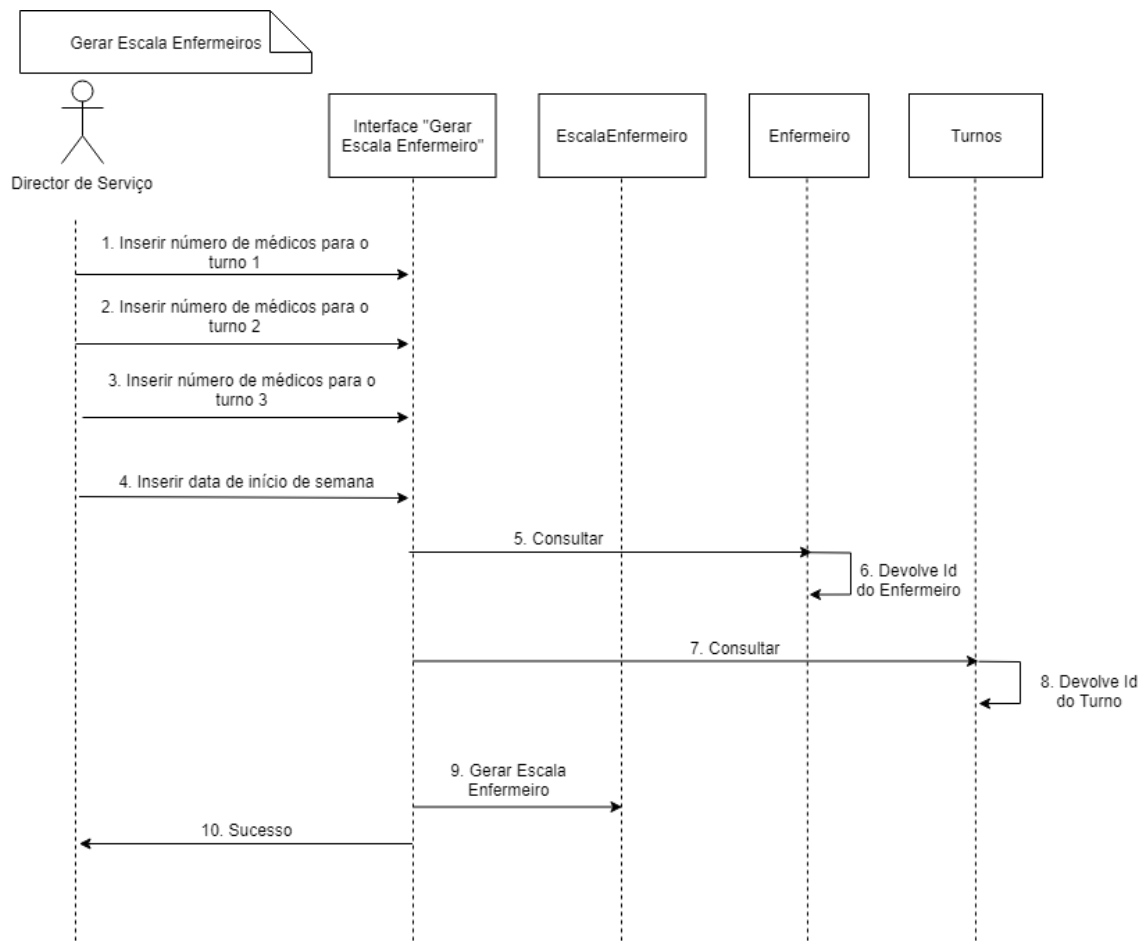


Figura 7 - Diagrama de Sequência Gerar escala enfermeiros



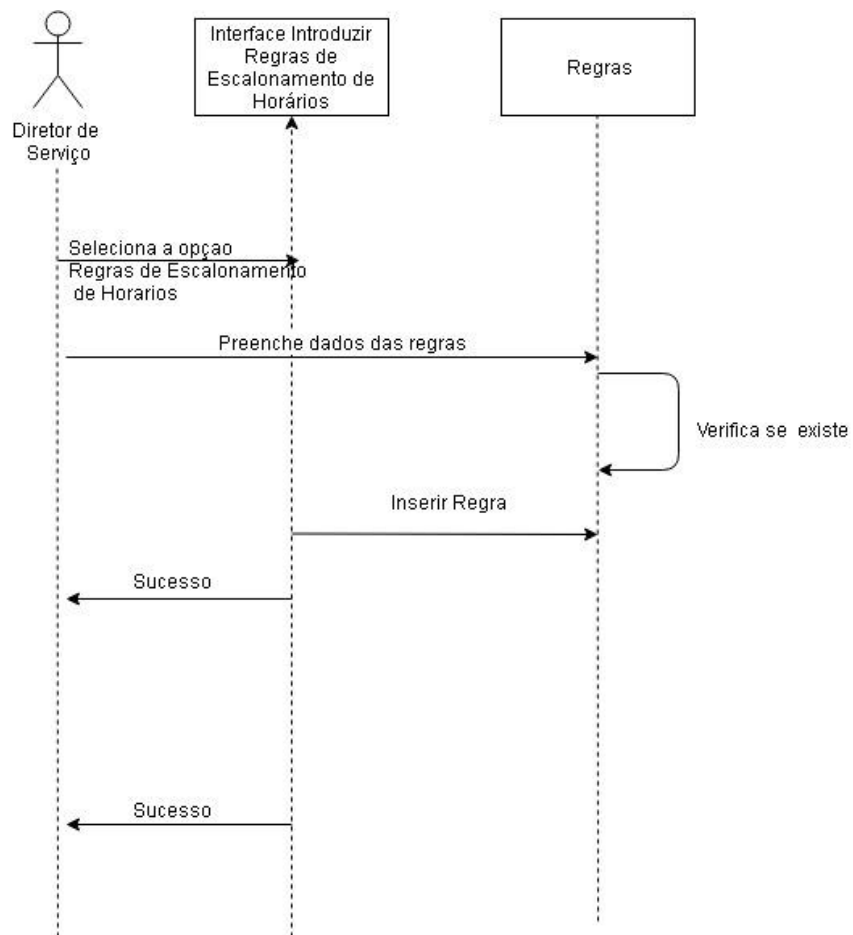


Figura 8 - Diagrama de Sequência Introduzir regras de escalonamento de horários

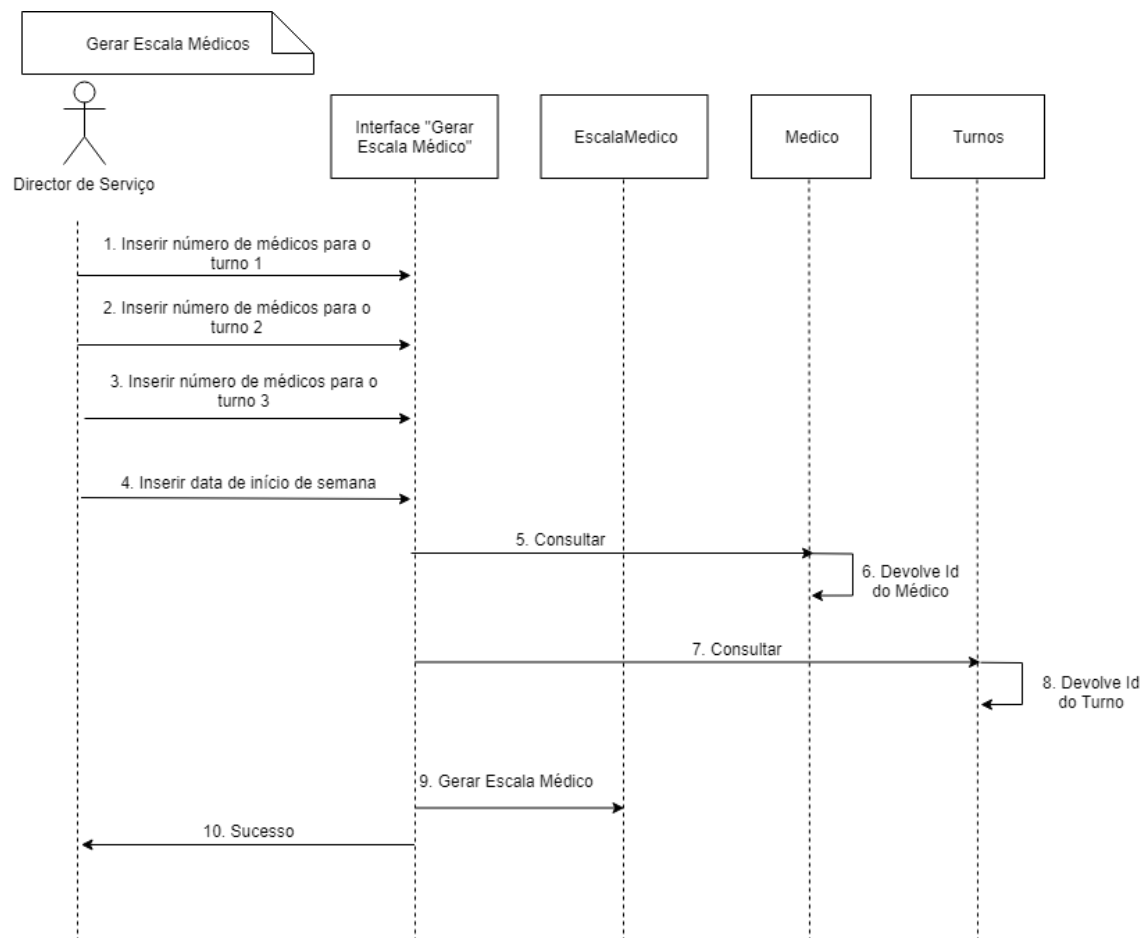


Figura 9 - Diagrama de Sequencia Gerar escala médico

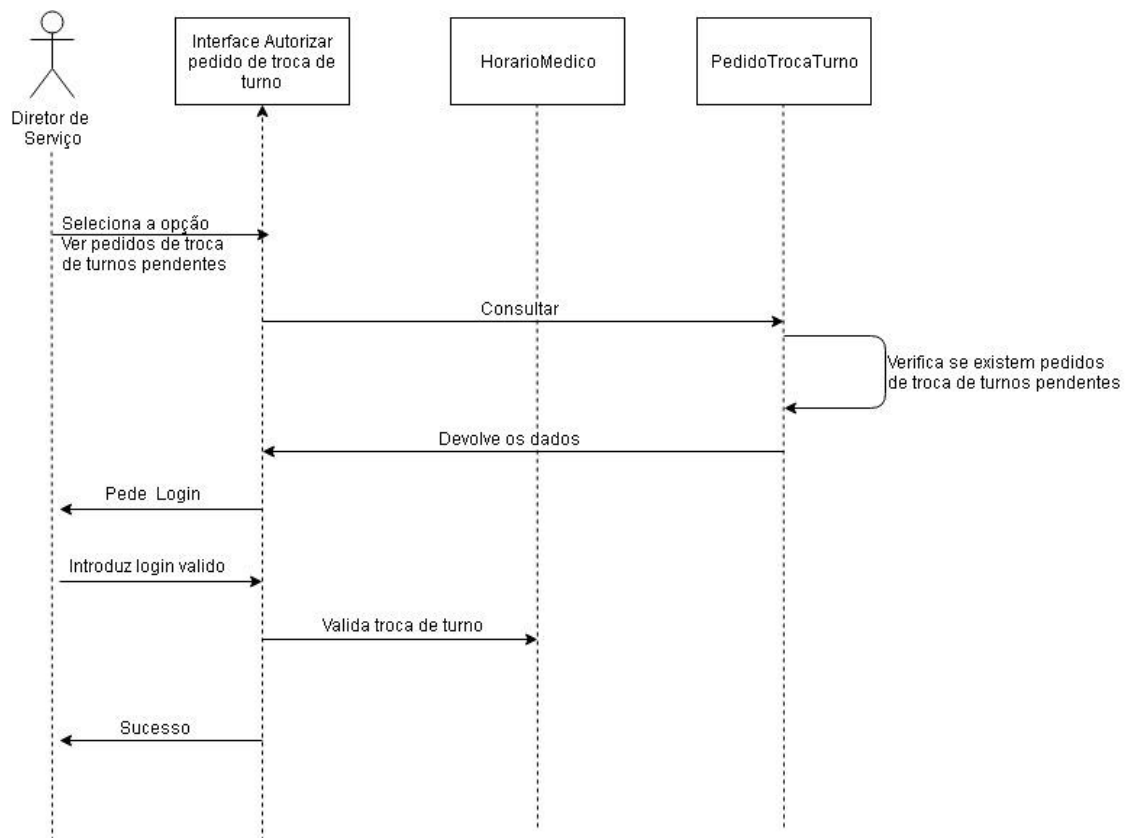


Figura 10 - Diagrama de Sequência Autorizar pedido de troca de turno

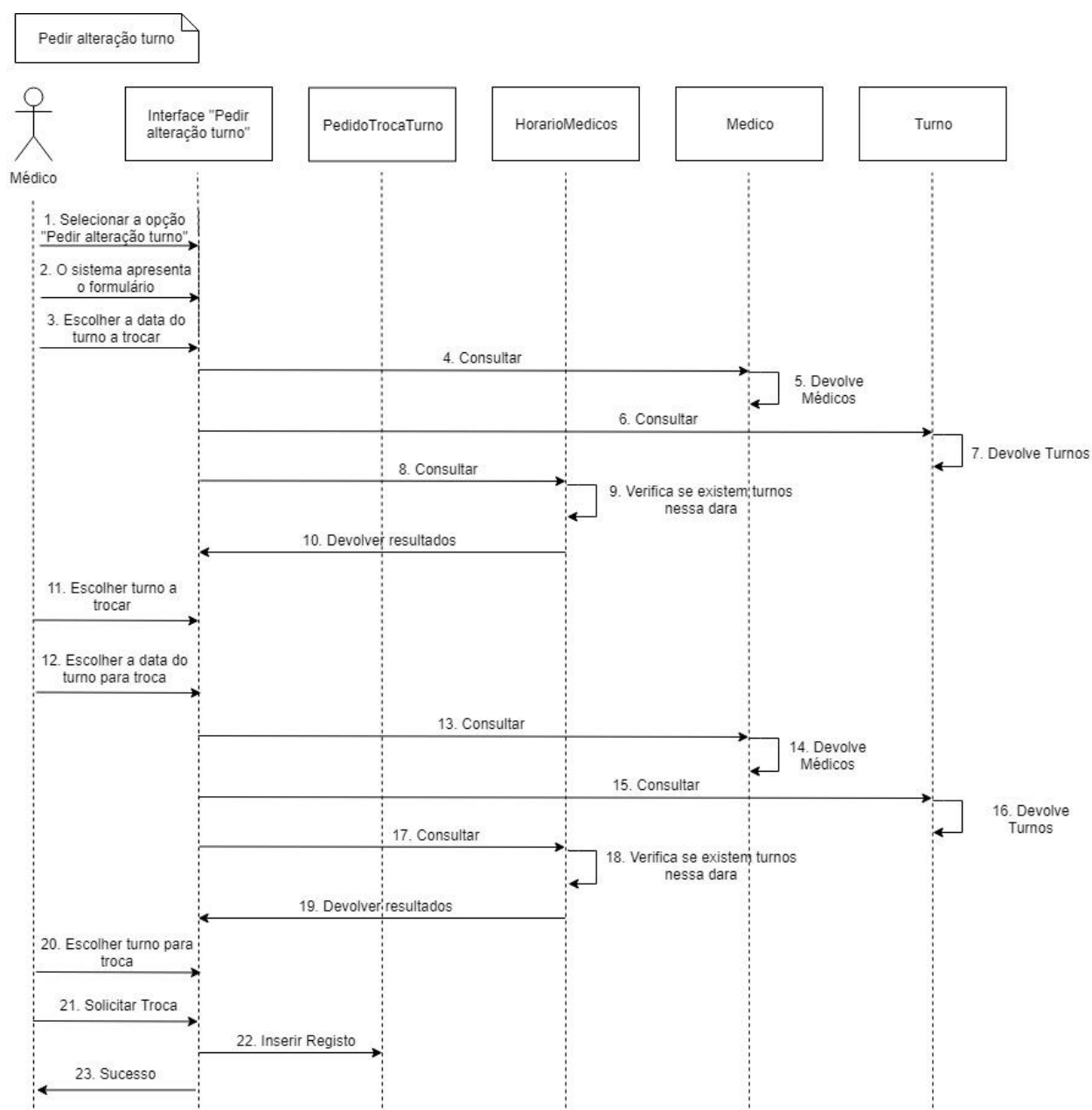


Figura 11 - Diagrama de Sequência "Pedir alteração turno"

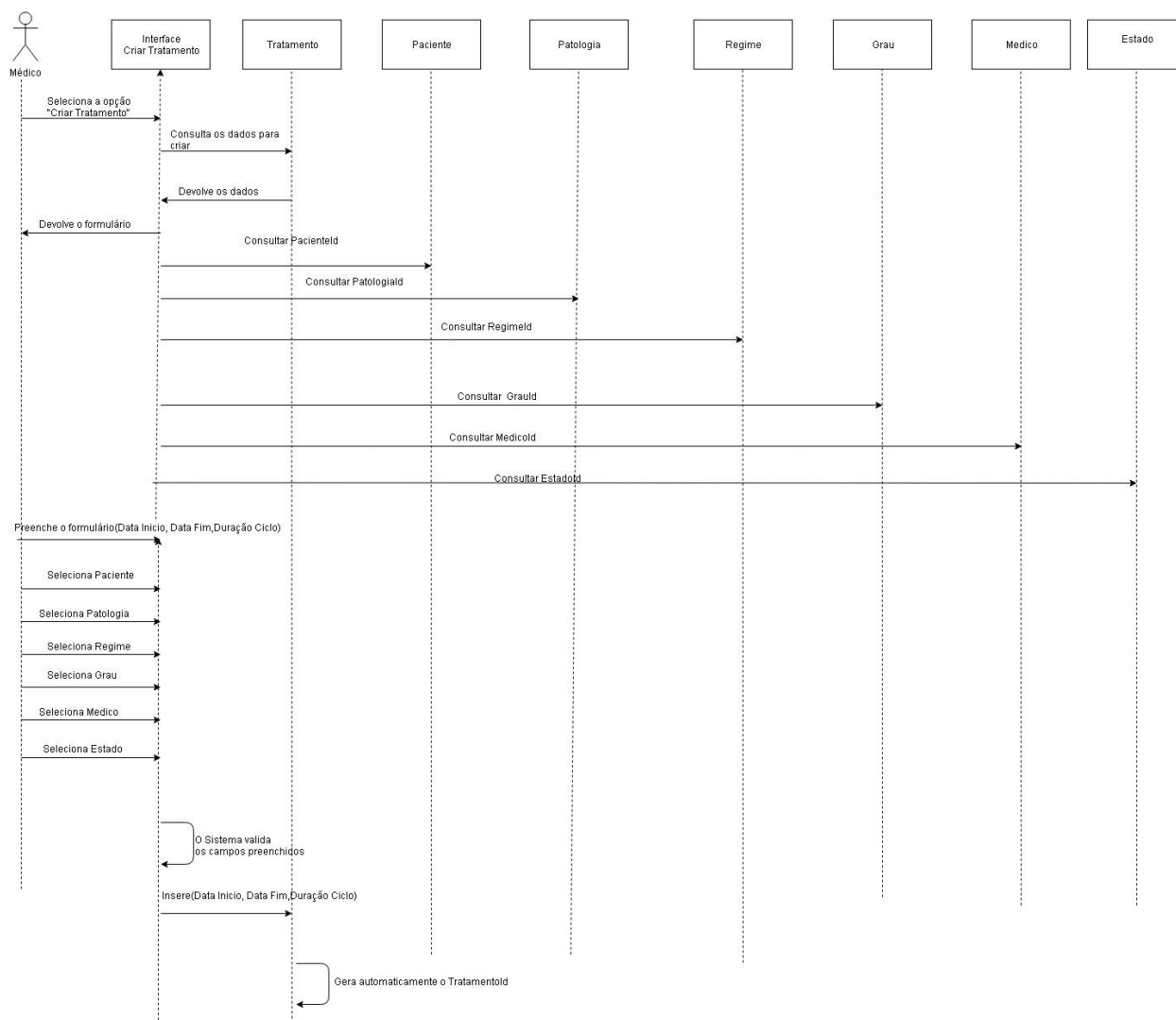


Figura 12 - Diagrama de Sequência Criar tratamento

# 5. Algoritmo

## 5.1 Algoritmo gerar escalas

INICIO

1. INICIAR a variável **numPessoasT1** (número de pessoas no turno 1) com o valor 5;
2. INICIAR a variável **numPessoasT2** (número de pessoas no turno 2) com o valor 3;
3. INICIAR a variável **numPessoasT3** (número de pessoas no turno 3) com o valor 2;
4. INICIAR a variável **segunda** com o valor 2;
5. INICIAR a variável **sexta** com o valor 6;
6. INICIAR o **arrayPessoa** com os respectivos ID's da pessoa inserida pelo utilizador (Ex.: médico ou enfermeiro).
7. INICIAR o **arrayPessoaSemFilhos** com os respectivos ID's das pessoas que não tiverem filhos.
8. INICIAR o **arrayPessoaComFilhos** com os respectivos ID's das pessoas que tiverem filhos.
9. DECLARAR a variável **pessoaT1**;
10. DECLARAR a variável **pessoaT2**;
11. DECLARAR a variável **pessoaT3**;
12. INICIAR a variável **ano** com o valor 2018;
13. INICIAR a variável **mes** com o valor 11;
14. INICIAR a variável **dia** com o valor 26;
15. DECLARAR a variável **data**;
16. INICIAR a variável **idPessoaT1** com o valor 0;
17. INICIAR a variável **idPessoaT2** com o valor 0;
18. INICIAR a variável **idPessoaT3** com o valor 0;
19. INICIAR a variável **nPessoasComFolga** com o valor **arrayPessoas.lenght - (numPessoasT1 + numPessoasT2 + numPessoasT3)**;
20. INICIAR o **arrayPessoaComFolga**;
21. DECLARAR o **arrayPessoaNoite**;
22. INICIAR a lista **listaPessoas** com o **arrayPessoa**
23. DECLARAR a lista **listaPessoasSemFilhos**
24. DECLARAR a lista **listaPessoasComFilhos**
25. PARA variável **i = segunda** ATÉ **i = sexta** FAZER

1. INICIAR a lista **listaPessoasSemFilhos** com o **arrayPessoaSemFilhos**
2. INICIAR a lista **listaPessoasComFilhos** com o **arrayPessoaComFilhos**
3. PARA variável **x = 0** ATÉ **x < nPessoasComFolga** FAZER
  1. COLOCAR na posição **x** do **arrayPessoaComFolga** o conteúdo da posição aleatória do **arrayPessoa**;
  2. REMOVER da **listaPessoas** o conteúdo da posição **x** do **arrayPessoaComFolga**
  3. REMOVER da **listaPessoasSemFilhos** o conteúdo da posição **x** do **arrayPessoaComFolga**
  4. REMOVER da **listaPessoasComFilhos** o conteúdo da posição **x** do **arrayPessoaComFolga**

FIM\_PARA

4. REMOVER da lista **listaPessoasSemFilhos** o conteúdo do **arrayPessoaNoite**
5. PARA variável **y = 0** ATÉ **y < nPessoasT1** FAZER
  1. DECLARAR a variável **turno = "MANHÃ"**
  2. SE **y** for PAR
  3. **idPessoaT1 = random(arrayPessoaComFilhos)**
  4. **data = (ano,mes,dia)**
  5. SENÃO
  6. **idPessoaT1 = random(arrayPessoaSemFilhos)**
  7. **data = (ano,mes,dia)**
  8. FIM\_SE
  9. CHAMAR a função **inserirEscalaPessoa(data.AddDays(i-2), turno, bd, idPessoaT1)**
  10. REMOVER da lista **listaPessoasComFilhos** o **idPessoaT1**
  11. REMOVER da lista **listaPessoasSemFilhos** o **idPessoaT1**

FIM\_PARA

```

6. ADICIONAR à lista listaPessoasSemFilhos o conteúdo do array
   arrayPessoaNoite

7. PARA variável z = 0 ATÉ z = nPessoasT2 FAZER
   1.DECLARAR a variável turno = "TARDE"
   2.SE z for PAR
   3.idPessoaT2 = random(arrayPessoaComFilhos)
   4.SENÃO
   5.idPessoaT2 = random(arrayPessoaSemFilhos)
   6.FIM_SE
   7.data = (ano,mes,dia)
   8.CHAMAR a função inserirEscalaPessoa (data.AddDays(i-2), turno, bd,
     idPessoaT2)
   9.REMOVER da lista listaPessoasSemFilhos o idPessoaT2
   10.REMOVER da lista listaPessoasSemFilhos o idPessoaT2
   FIM_PARA

8. INICIAR o arrayPessoaNoite;

9. PARA variável k = 0 ATÉ k < nPessoasT3 FAZER
   1.DECLARAR a variável turno = "NOITE"
   2.idPessoaT3 = random(arrayPessoaSemFilhos)
   3.COLOCAR na posição k do arrayPessoaNoite o conteúdo da variável
     idPessoaT3
   4.data = (ano,mes,dia+1)
   5.CHAMAR a função inserirEscalaPessoa (data.AddDays(i-2), turno, bd,
     idPessoaT3)
   6.REMOVER da lista listaPessoasSemFilhos o idPessoaT3
   FIM_PARA
FIM_PARA

```

FIM

## 5.2 Algoritmo gerar horários

INICIO

```

26. INICIAR a variável numPessoasT1 (número de pessoas no turno 1) com o valor 5;
27. INICIAR a variável numPessoasT2 (número de pessoas no turno 2) com o valor 3;
28. INICIAR a variável numPessoasT3 (número de pessoas no turno 3) com o valor 2;
29. INICIAR a variável segunda com o valor 2;
30. INICIAR a variável sexta com o valor 6;
31. INICIAR o arrayPessoa com os respectivos ID's da pessoa inserida pelo utilizador
   (Ex.: médico ou enfermeiro).
32. INICIAR o arrayPessoaSemFilhos com os respectivos ID's das pessoas que não
   tiveram filhos.
33. INICIAR o arrayPessoaComFilhos com os respectivos ID's das pessoas que tiveram
   filhos.
34. DECLARAR a variável pessoaT1;
35. DECLARAR a variável pessoaT2;
36. DECLARAR a variável pessoaT3;
37. INICIAR a variável ano com o valor 2018;
38. INICIAR a variável mes com o valor 11;
39. INICIAR a variável dia com o valor 26;
40. DECLARAR a variável data;
41. INICIAR a variável idPessoaT1 com o valor 0;
42. INICIAR a variável idPessoaT2 com o valor 0;
43. INICIAR a variável idPessoaT3 com o valor 0;
44. INICIAR a variável nPessoasComFolga com o valor arrayPessoas.lenght -
   (numPessoasT1 + numPessoasT2 + numPessoasT3);
45. INICIAR o arrayPessoaComFolga;
46. DECLARAR o arrayPessoaNoite;
47. INICIAR a lista listaPessoas com o arrayPessoa
48. DECLARAR a lista listaPessoasSemFilhos
49. DECLARAR a lista listaPessoasComFilhos
50. PARA variável i = segunda ATÉ i = sexta FAZER

   1. INICIAR a lista listaPessoasSemFilhos com o arrayPessoaSemFilhos
   2. INICIAR a lista listaPessoasComFilhos com o arrayPessoaComFilhos

```

```

3. PARA variável x = 0 ATÉ x < nPessoasComFolga FAZER
    1.COLOCAR na posição x do arrayPessoaComFolga o conteúdo da
      posição aleatória do arrayPessoa;
    2.REMOVER da listaPessoas o conteúdo da posição x do
      arrayPessoaComFolga
    3.REMOVER da listaPessoasSemFilhos o conteúdo da posição x do
      arrayPessoaComFolga
    4.REMOVER da listaPessoasComFilhos o conteúdo da posição x do
      arrayPessoaComFolga
    FIM_PARA

4. REMOVER da lista listaPessoasSemFilhos o conteúdo do array
   arrayPessoaNoite

5. PARA variável y = 0 ATÉ y < nPessoasT1 FAZER
    1.DECLARAR a variável turno = "MANHÃ"
    2.SE y for PAR
    3.idPessoaT1 = random(arrayPessoaComFilhos)
    4.data = (ano,mes,dia,9,0,0)
    5.SENÃO
    6.idPessoaT1 = random(arrayPessoaSemFilhos)
    7.data = (ano,mes,dia,8,0,0)
    8.FIM_SE
    9.CHAMAR a função inserirHorarioPessoa(data.AddDays(i-2), turno, bd,
      idPessoaT1)
    10. REMOVER da lista listaPessoasComFilhos o idPessoaT1
    11. REMOVER da lista listaPessoasSemFilhos o idPessoaT1
    FIM_PARA

6. ADICIONAR à lista listaPessoasSemFilhos o conteúdo do array
   arrayPessoaNoite

7. PARA variável z = 0 ATÉ z = nPessoasT2 FAZER
    1.DECLARAR a variável turno = "TARDE"
    2.SE z for PAR
    3.idPessoaT2 = random(arrayPessoaComFilhos)
    4.SENÃO
    5.idPessoaT2 = random(arrayPessoaSemFilhos)
    6.FIM_SE
    7.data = (ano,mes,dia,16,0,0)
    8.CHAMAR a função inserirHorarioPessoa (data.AddDays(i-2), turno,
      bd, idPessoaT2)
    9.REMOVER da lista listaPessoasSemFilhos o idPessoaT2
    10. REMOVER da lista listaPessoasSemFilhos o idPessoaT2
    FIM_PARA

8. INICIAR o arrayPessoaNoite;

9. PARA variável k = 0 ATÉ k < nPessoasT3 FAZER
    1.DECLARAR a variável turno = "NOITE"
    2.idPessoaT3 = random(arrayPessoaSemFilhos)
    3.COLOCAR na posição k do arrayPessoaNoite o conteúdo da variável
      idPessoaT3
    4.data = (ano,mes,dia+1,0,0,0)
    5.CHAMAR a função inserirHorarioPessoa (data.AddDays(i-2), turno,
      bd, idPessoaT3)
    6.REMOVER da lista listaPessoasSemFilhos o idPessoaT3
    FIM_PARA
FIM_PARA

```

FIM



## 6. Protótipos

