36513 Bianca do Santos Marinho

40639 Jean Michel Vaz Luth

**Funcionalitester**

BRASÍLIA, DF

2017

36513 Bianca do Santos Marinho

40639 Jean Michel Vaz Luth

**Funcionalitester**

Projeto da disciplina Projeto Integrador: Projeto de Sistemas, do Centro Universitário Unieuro, de Brasília, DF.

BRASÍLIA, DF

2017

36513 Bianca do Santos Marinho

40639 Jean Michel Vaz Luth

**Funcionalitester**

**BANCA EXAMINADORA - APROVADO POR:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<titulação e nome >

Centro Universitário Unieuro, DF

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<titulação e nome>

Centro Universitário Unieuro, DF

Brasília, DF

<2017>

**RESUMO**

Este projeto tem como objetivo apresentar a realidade de registros de teste realizados sob um desenvolvimento de software e uma solução para trabalho em equipe juntamente aos analistas e programadores.

Palavras-chaves: Software, Teste, Desenvolvedor.

**LISTA DE FIGURAS**

**Figura 1: Diagrama Atividades de Negócio**

**Figura 2: Modelo de Casos de Uso**

**Figura 3: Diagrama de Classes**

**Figura 4: Diagrama de Sequência – Caso de uso 01**

**Figura 5: Diagrama de Sequência – Caso de uso 01**

**Figura 6: Diagrama de Sequência – Caso de uso 01**

**Figura 7: Diagrama de Sequência – Caso de uso 02**

**Figura 8: Diagrama de Sequência – Caso de uso 02**

**Figura 9: Diagrama de Sequência – Caso de uso 02**

**Figura 10: Diagrama de Sequência – Caso de uso 03**

**Figura 11: Diagrama de Sequência – Caso de uso 03**

**Figura 12: Diagrama de Sequência – Caso de uso 03**

**Figura 13: Diagrama de Sequência – Caso de uso 04**

**SUMÁRIO**

[**1**](#_1hmsyys) **INTRODUÇÃO 10**

[*1.1*](#_41mghml) *Motivação 10*

[*1.2*](#_2grqrue) *Objetivo 10*

[**2**](#_vx1227) **VISÃO DO NEGÓCIO 10**

[*2.1*](#_3fwokq0) *Elementos do Negócio 10*

[*2.2*](#_1t3h5sf) *Modelo de Negócio 10*

[**3**](#_1v1yuxt) **VISÃO DE CASOS DE USO 11**

[*3.1*](#_4f1mdlm) *Modelo de Casos de Uso 11*

[*3.2*](#_2u6wntf) *Documentação dos Casos de Uso 12*

[UC01.](#_147n2zr) <Estilo: Caso de Uso. Manter Software> 13

UC02 <Estilo: Caso de Uso. Manter Funcionalidade> 13

UC03 <Estilo: Caso de Uso. Manter Teste> 15

UC04 <Estilo: Caso de Uso. Atualizar Ocorrência> 16

[**4**](#_19c6y18) **VISÃO DE DOMÍNIO 16**

[*4.1*](#_3tbugp1) *Dicionário de classes de domínio 16*

[*4.2*](#_28h4qwu) *Diagrama de Classes de Domínio 18*

5**VISÃO DE ESTADO E COMPORTAMENTO 20**

UC01. <Estilo: Sequência. Manter Software\_Cadastrar Software> 20

UC01. <Estilo: Sequência. Manter Software\_Alterar Software> 21

UC01. <Estilo: Sequência. Manter Software\_Excluir Software> 22

UC02. <Estilo: Sequência. Manter Funcionalidade\_Cadastrar Funcionalidade> 22

UC02. <Estilo: Sequência. Manter Funcionalidade\_Alterar Funcionalidade> 24

UC02. <Estilo: Sequência. Manter Funcionalidade\_Apagar Funcionalidade> 25

UC03. <Estilo: Sequência. Manter Teste\_Cadastrar Teste> 25

UC03. <Estilo: Sequência. Manter Teste\_Alterar Teste> 26

UC03. <Estilo: Sequência. Manter Teste\_Apagar Teste> 27

UC04. <Estilo: Sequência. Atualizar Ocorrência> 28

**6****REFERÊNCIAS BIBLIOGRÁFICAS 29**

1. **INTRODUÇÃO**
   1. ***Motivação***

A equipe foi motivada por um tema proposto pelo professor da disciplina e à pedido de uma professora que ministra na aula de Teste.

* 1. ***Objetivo***

O projeto tem como objetivo apresentar uma solução para as áreas de teste em conjunto com desenvolvimento, de forma que traz uma melhor comunicação com todos envolvidos no desenvolvimento de um software e melhorando a sua qualidade.

1. **VISÃO DO NEGÓCIO**
   1. ***Elementos do Negócio***

Em uma forma bem manual, o testador registra seus testes em um software ou em uma planilha, geralmente ambos isoladamente dos registros do analista e do desenvolvedor, fazendo com que dessa forma, tanto o desenvolvedor ou analista passam a ter conhecimento dos testes quando o testador lhes apresentam em um relatório ou algo do tipo.

* 1. ***Modelo de Negócio***

O diagrama a seguir representa a funcionalidade de todo o sistema, onde o sistema exige identificação para qualquer um dos usuários. Inicialmente se faz a identificação do usuário para que o mesmo realize a tarefa desejada e de acordo com a sua função, começando pelo analista que deve fazer registro do software e suas funcionalidades para que em seguida, o testador faça o registro dos testes realizados “naquela” funcionalidade, e em seguida, o desenvolvedor tem acesso a esses testes realizados para que o mesmo possa ter as informações necessárias para realização da manutenção.

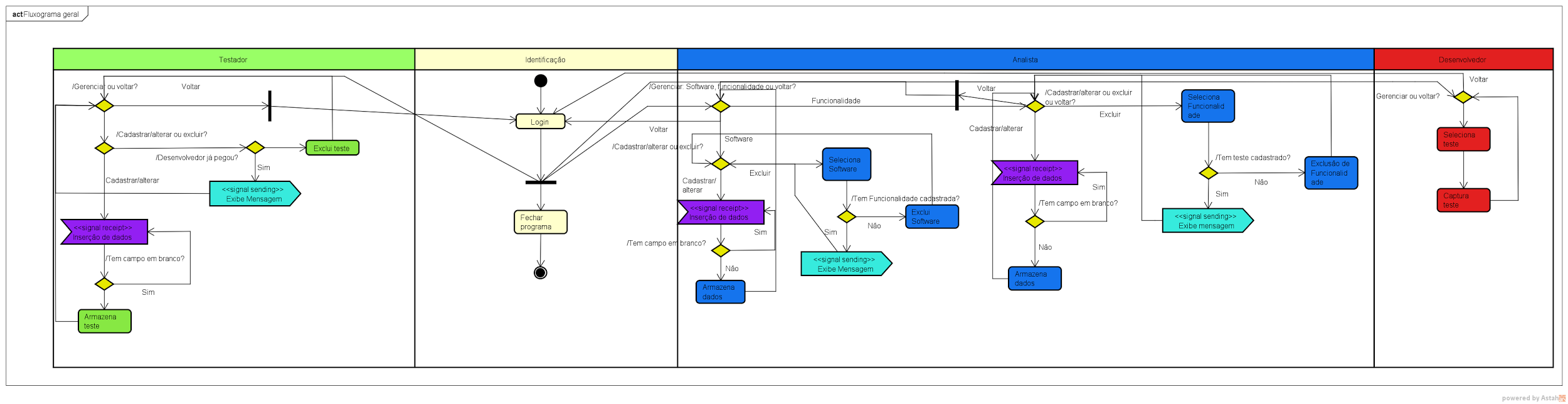


Figura 1:Diagrama Atividades de Negócio

Fonte: Elaborada pelo autor

1. **VISÃO DE CASOS DE USO**
   1. ***Modelo de Casos de Uso***

-Analista de Sistema: O analista tem como objetivo gerenciar os softwares e funcionalidades (cadastro, alteração e exclusão).

-Testador: O testador tem como objetivo gerenciar os testes, através do softwares e funcionalidades cadastradas pelo analista.

-Desenvolvedor: O desenvolver tem como objetivo consultar as ocorrências.

A Figura 2 representa o modelo de casos de uso do sistema.

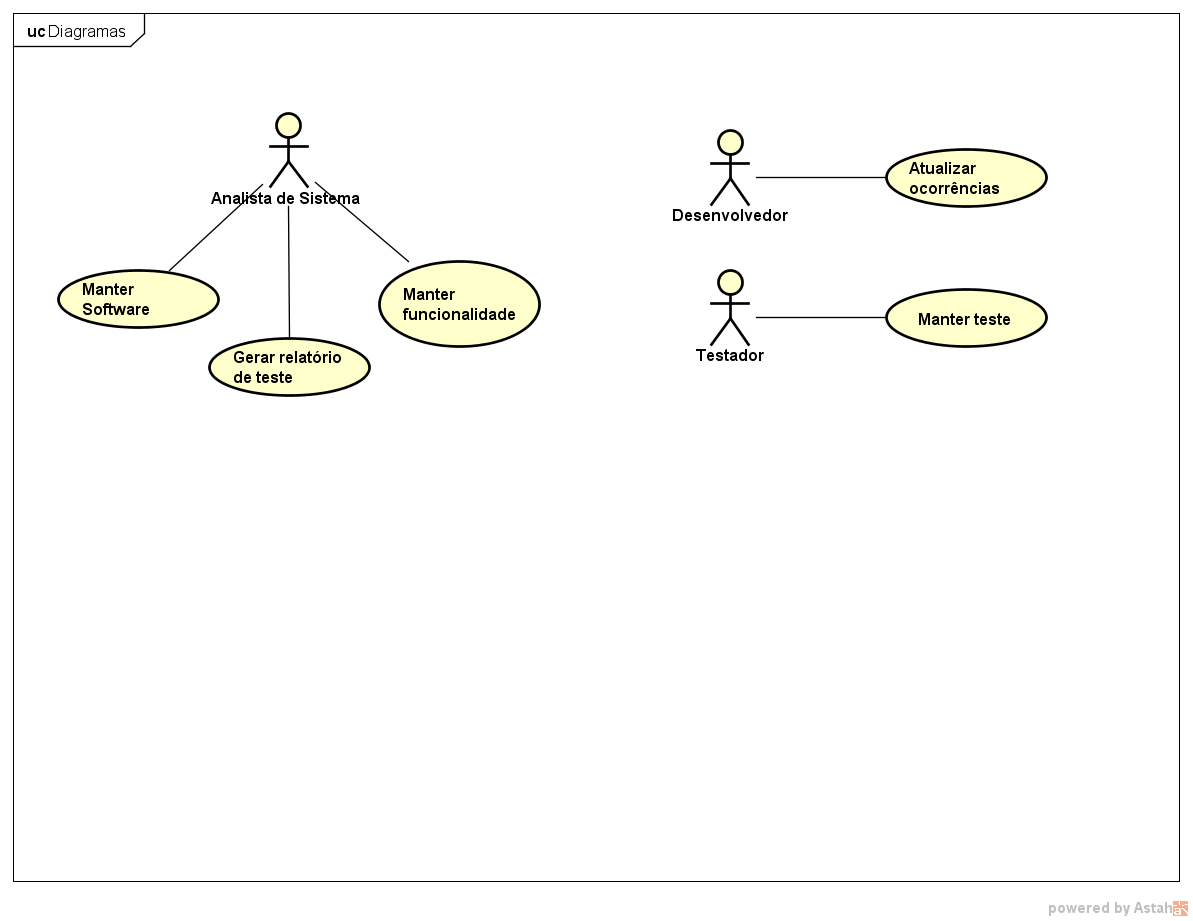


Figura 2: Modelo de Caso de Uso

Fonte: Elaborada pelo autor

* 1. ***Documentação dos Casos de Uso***

|  |  |
| --- | --- |
| 1. **Caso de Uso. Manter Software** | |
| Descreve a ação de cadastrar, alterar e excluir software. | |
| Fluxo Principal | 1. O usuário acessa o sistema 2. O sistema exibe a tela inicial com opção para cadastrar, alterar ou excluir software 3. Usuário busca o serviço/produto desejado na caixa de pesquisa do sistema. 4. Sistema retorna ao usuário a ação que o usuário selecionou. |
| Fluxos Alternativos | Caso haja uma funcionalidade cadastrada, o sistema informa que a funcionalidade deve ser apagada para que o software possa ser apagado. |
| Regras de Negócio | Caso haja alguma funcionalidade cadastrada em um software, o analista não pode fazer ou exclusão do mesmo até que as funcionalidades estejam apagadas. |
| Requisitos de Interface | É necessário que haja analista cadastrado. |

|  |  |
| --- | --- |
| 1. **Caso de Uso. Manter funcionalidade** | |
| Descreve a ação de realizar o cadastro, alteração e exclusão de funcionalidade | |
| Fluxo Principal | 1. O requerente acesso o sistema 2. O Sistema exibe a tela inicial com opção para cadastrar, alterar ou excluir funcionalidade 3. Usuário busca o serviço desejado na caixa de pesquisa do sistema. 4. Sistema retorna ao usuário a ação que o usuário selecionou. |
| Fluxos Alternativos | Caso haja um teste cadastrado o sistema retorna uma mensagem ao usuário exibindo que o analista não pode fazer a exclusão do mesmo até que as funcionalidades estejam apagadas. |
| Regras de Negócio | **R.N 1-** Não pode fazer exclusão enquanto um teste daquela funcionalidade estiver cadastrada.  **R.N 2-** Para realizar cadastro é necessário que tenha um software cadastrado.  **R.N 3-** Não é possível cadastrada funcionalidade se tiver opções em branco. |
| Requisitos de Interface | Para ter acesso a interface de funcionalidade, é necessário que haja cadastro de um analista e para sua total função é necessário que haja software cadastrado. |

|  |  |
| --- | --- |
| 1. **Caso de Uso. Manter teste** | |
| Descreve a ação de realizar o cadastro, alteração e exclusão de teste | |
| Fluxo Principal | 1. O testador abre o sistema. 2. O sistema exibe a página de teste. 3. O visitante clica em “Cadastre-se”, “Alterar” ou “Excluir” 4. Caso deseja cadastrar, o sistema exibe espaços que devem ser preenchidas pelo testador (área, ocorrência, tipo, prioridade, situação, observação). 5. Caso deseja alterar, o testador seleciona o teste, e altera o(s) campo(s) desejado(s). 6. O sistema registra os dados do cliente |
| Fluxos Alternativos | O sistema exibe uma mensagem para o testador de que o teste já foi capturado por algum desenvolvedor caso deseja realizar exclusão. |
| Regras de Negócio | **RN 1 -** O testador só pode realizar cadastro se houver funcionalidade cadastrada.  **RN 2** - O testador só pode apagar teste se nenhum desenvolvedor tiver pego o teste para manutenção. |
| Requisitos de Interface | Para ter acesso a interface é necessário que haja cadastro de testador e que haja funcionalidade e software cadastrado. |

|  |  |
| --- | --- |
| 1. **Caso de Uso. Atualizar ocorrência** | |
| Descreve a ação de capturar o teste para realizar manutenção | |
| Fluxo Principal | 1. O desenvolvedor abre o sistema 2. O sistema exibe os testes cadastrados. 3. O desenvolvedor captura o teste desejado. |
| Fluxos Alternativos |  |
| Regras de Negócio | **RN 1 -** O desenvolvedor só pode capturar teste que nenhum outro desenvolvedor capturou. |
| Requisitos de Interface | É necessário que haja teste cadastrado e desenvolvedor cadastrado. |

1. **VISÃO DE INTERAÇÃO DE OBJETOS**

**4.1 Dicionário de classes de domínio**

As classes, segundo o diagrama representado na Figura 3, foram

definidas como:

-Pessoa: Descreve as Pessoa que serão cadastrados no sistema.

Tem como métodos adicionar e alterar. A classe Pessoa está agregada com as classes Analista de Sistemas, Testador e Desenvolvedor.

-Analista de Sistema: O analista tem a função de cadastrar software e suas funcionalidades no sistema.

Tem como métodos cadastrar, alterar e excluir.

-Testador: O testador tem a função de realizar os testes e cadastrar os testes realizados no sistema.

Tem como métodos cadastrar, alterar e excluir.

-Desenvolvedor: O desenvolvedor tem a função de atualizar as ocorrências dos testes.

Tem como métodos pegar os testes para atualizar as ocorrências.

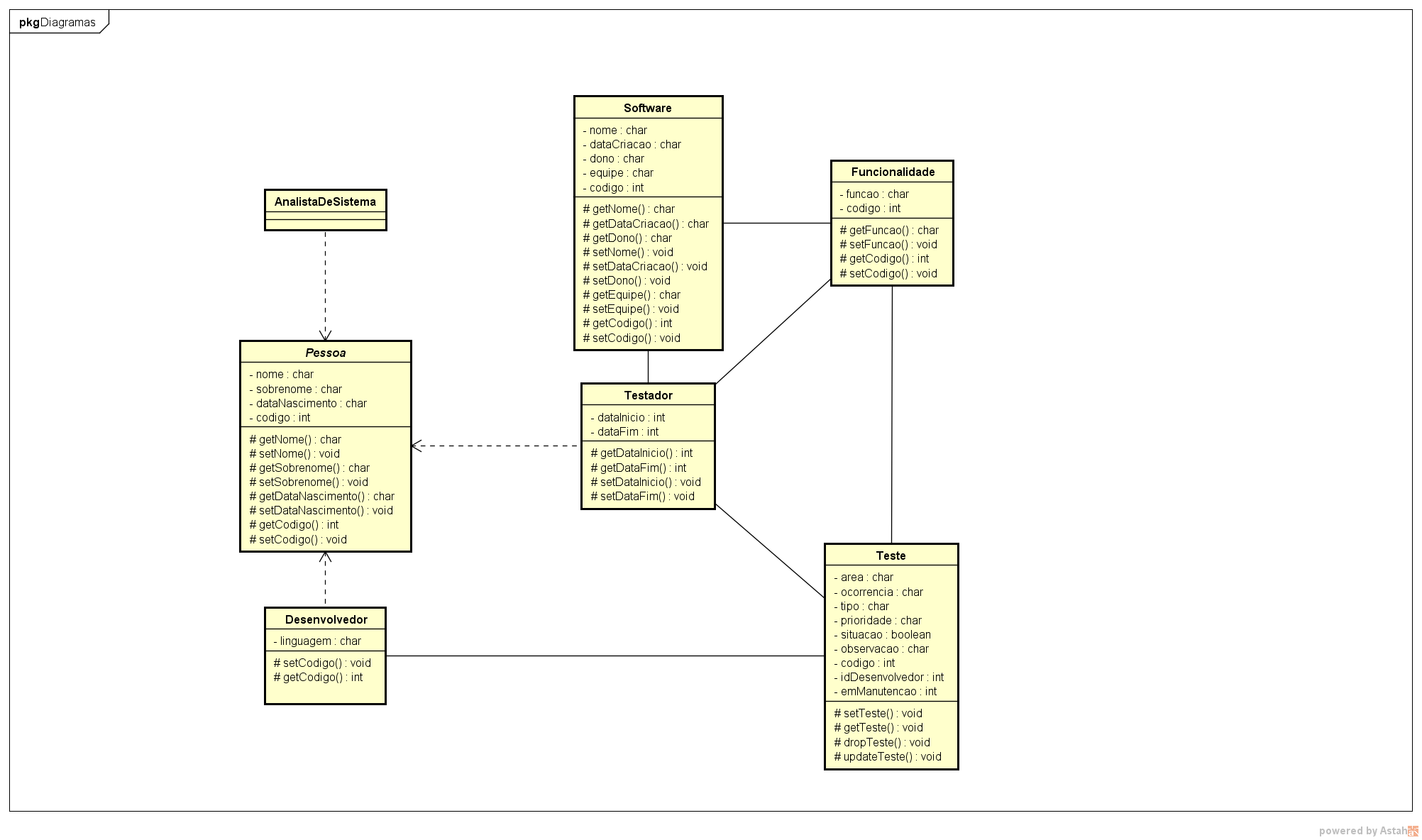
-Software: O registro de software é necessário para registrar as suas funcionalidades afim de registrar seus testes.

Tem como métodos registrar, atualizar e excluir.

-Teste: O teste é realizado pelo testador e seus resultados são armazenados para que o desenvolvedor tome ciência e possa resolvê-los caso haja algum problema.

**4.2 Diagrama de Classes de Domínio**

A Figura 3 representa o diagrama de classes referente ao sistema.

Figura 3: Diagrama de Classes

Fonte: Elaborada pelo autor

1. **VISÃO DE ESTADO E COMPORTAMENTO**

Neste tópico serão apresentados os diagramas de sequência

relacionados a cada caso de uso. O Diagrama de Sequência é uma das

ferramentas da UML usadas para representar interações entre objetos de um

cenário, realizadas através de operações ou métodos.

**5.1 UC01 Caso de Uso. Manter Software**

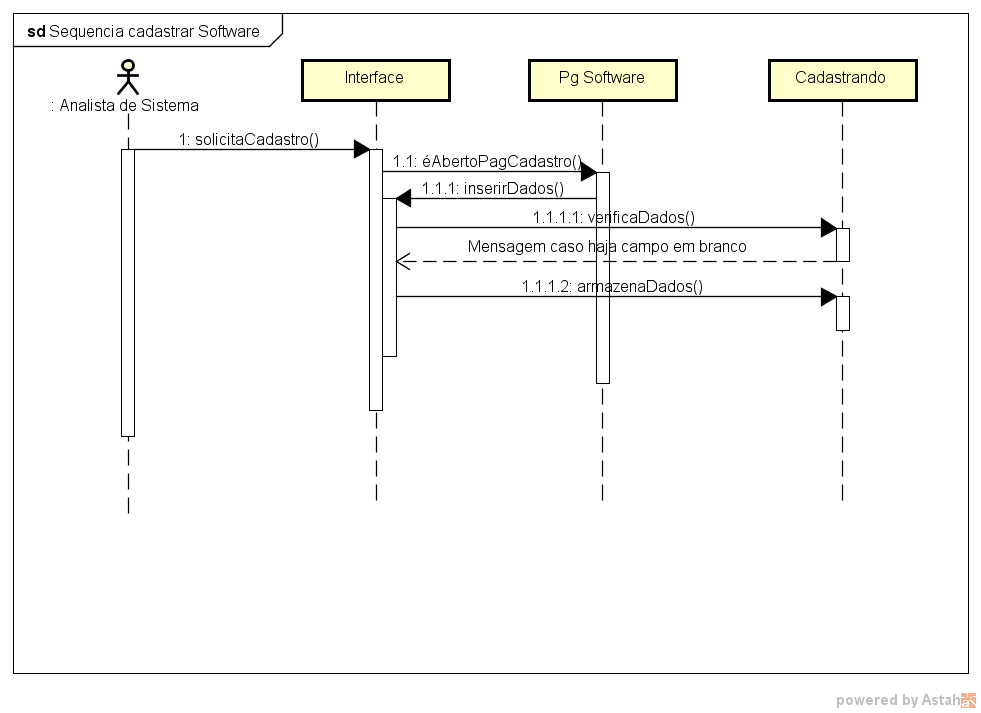
****

Figura 4: Diagrama de Sequência – Caso de uso 01

Fonte: Elaborada pelo autor

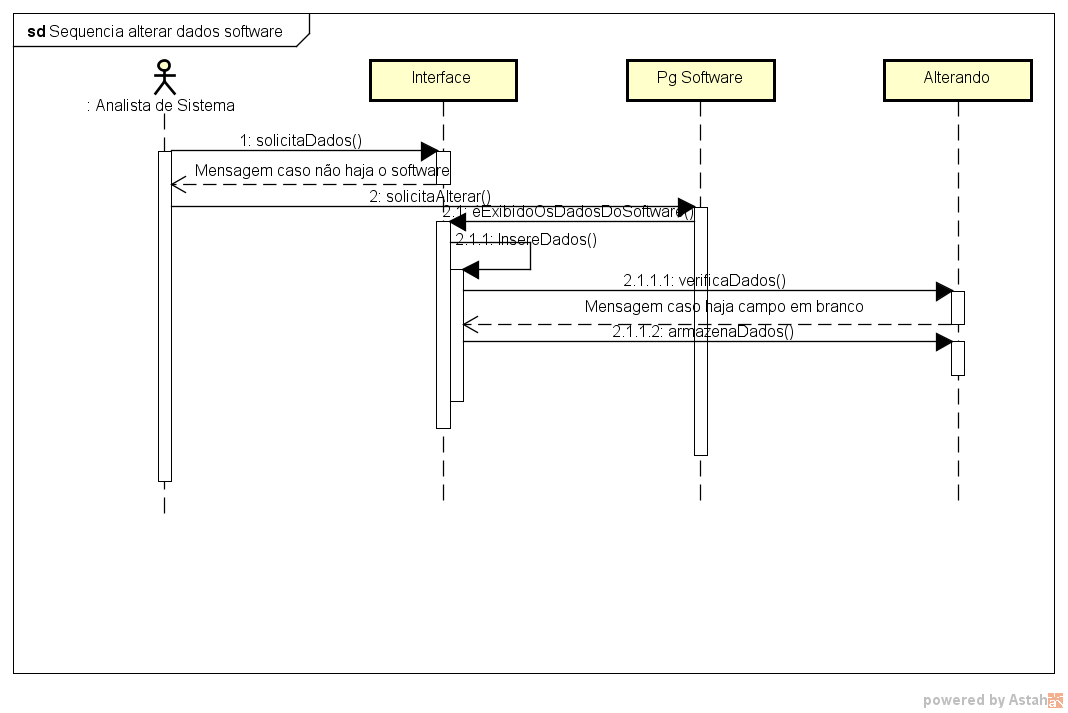


Figura 5: Diagrama de Sequência – Caso de uso 01

Fonte: Elaborada pelo autor

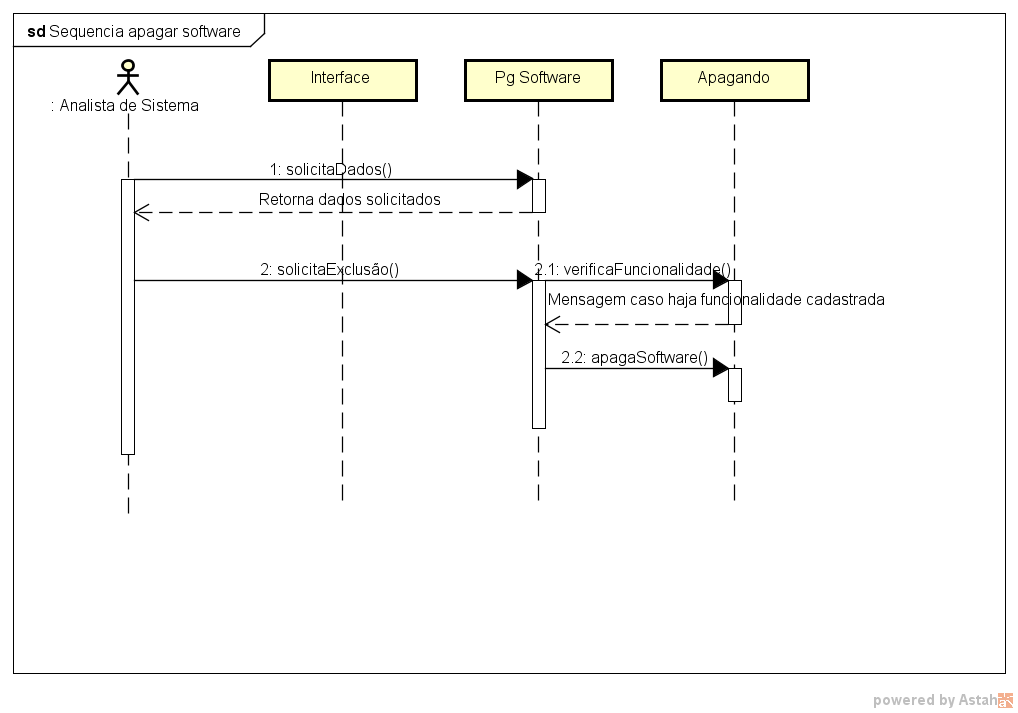


Figura 6: Diagrama de Sequência – Caso de uso 01

Fonte: Elaborada pelo autor

**5.2 UC02 Caso de Uso. Manter funcionalidade**

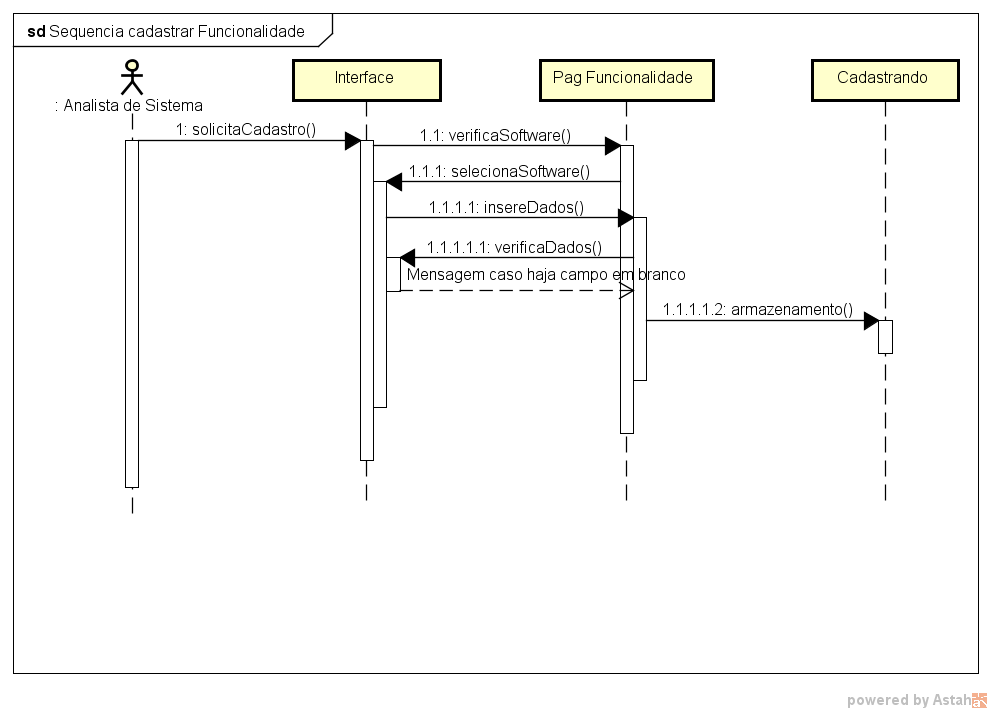
****

Figura 7: Diagrama de Sequência – Caso de uso 02

Fonte: Elaborada pelo autor

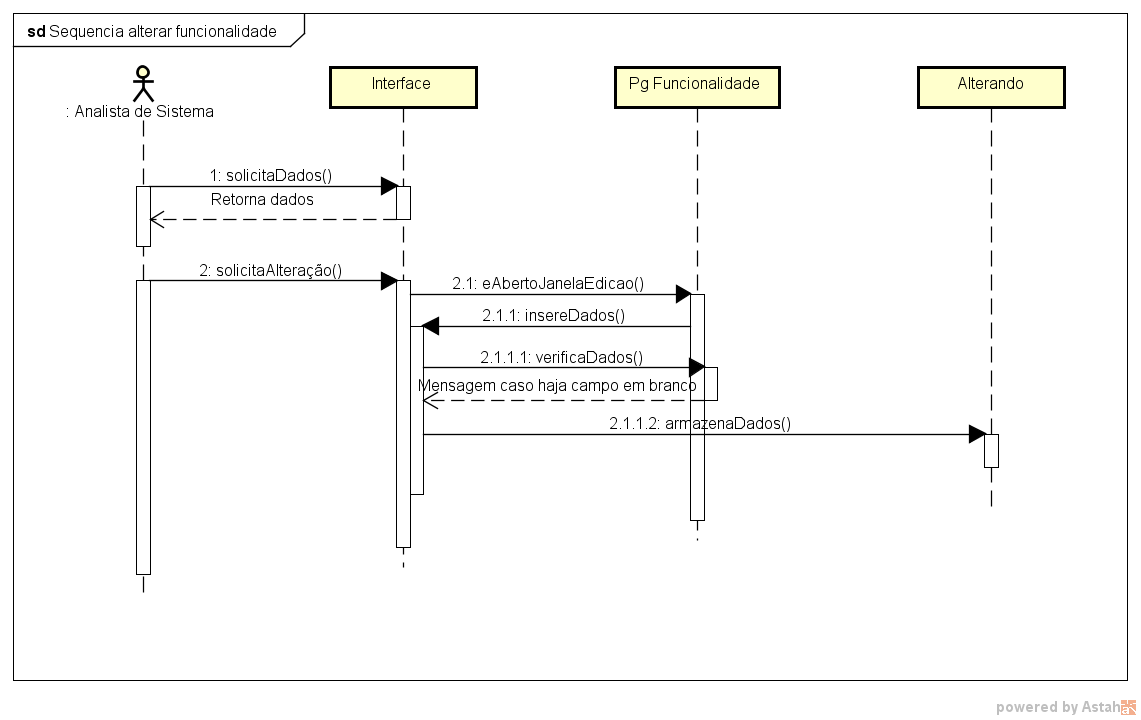


Figura 8: Diagrama de Sequência – Caso de uso 02

Fonte: Elaborada pelo autor

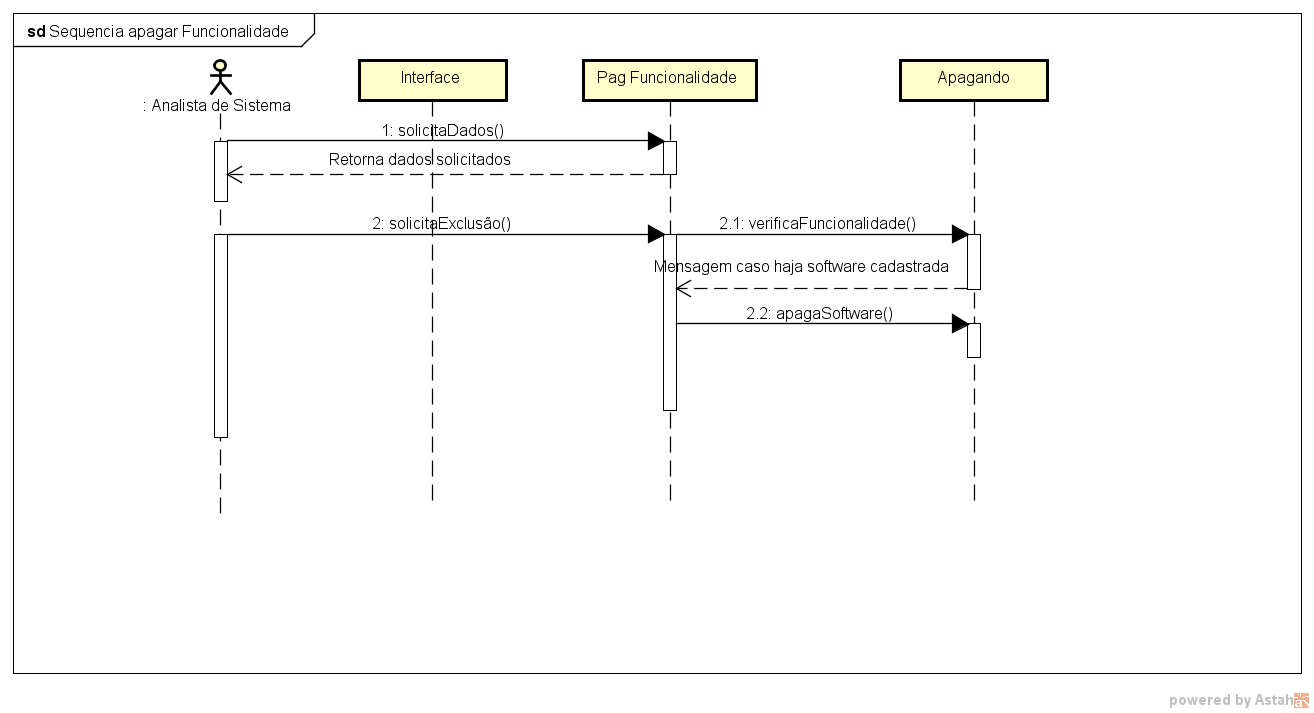
****

Figura 9: Diagrama de Sequência – Caso de uso 02

Fonte: Elaborada pelo autor

**5.3 UC03 Caso de Uso. Manter teste**

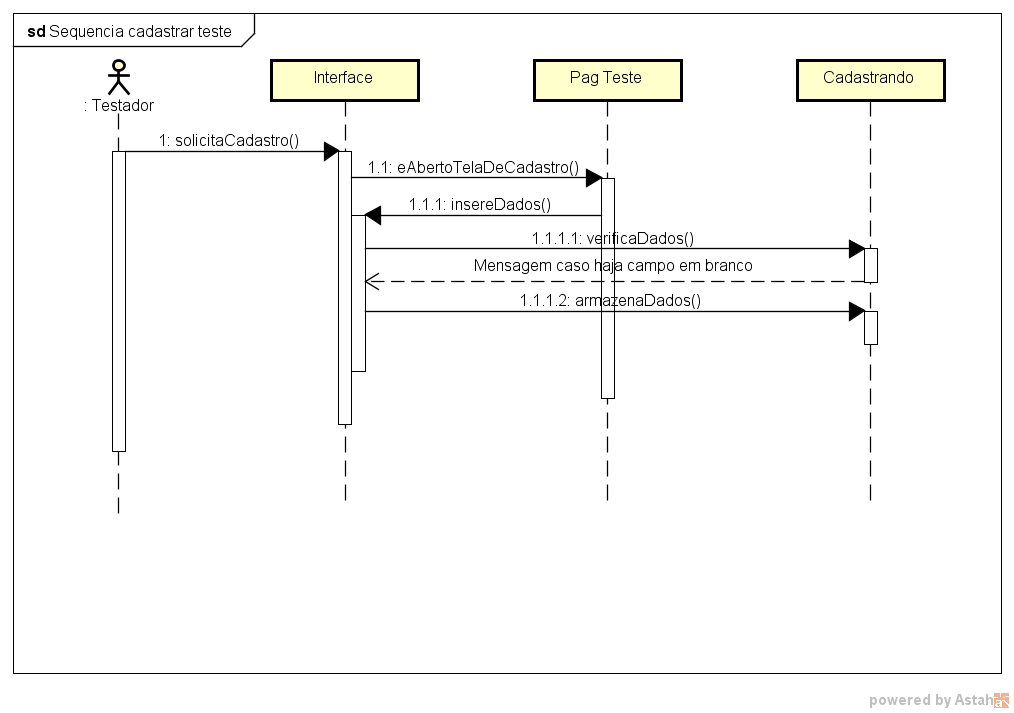
****

Figura 10: Diagrama de Sequência – Caso de uso 03

Fonte: Elaborada pelo autor

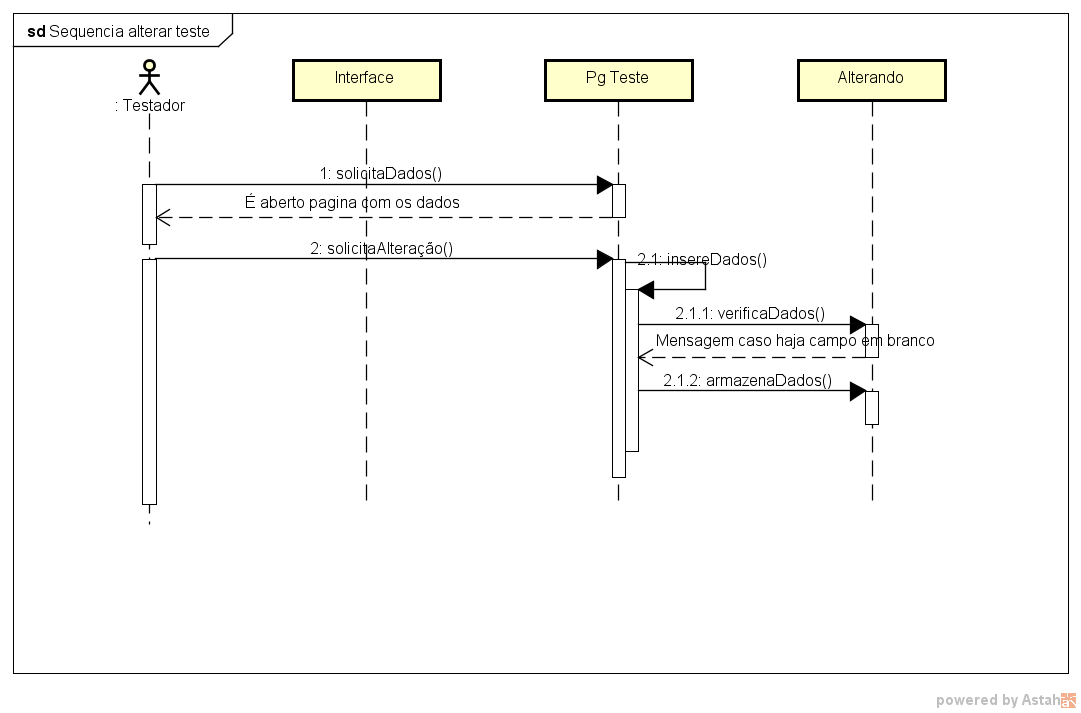


Figura 11: Diagrama de Sequência – Caso de uso 03

Fonte: Elaborada pelo autor

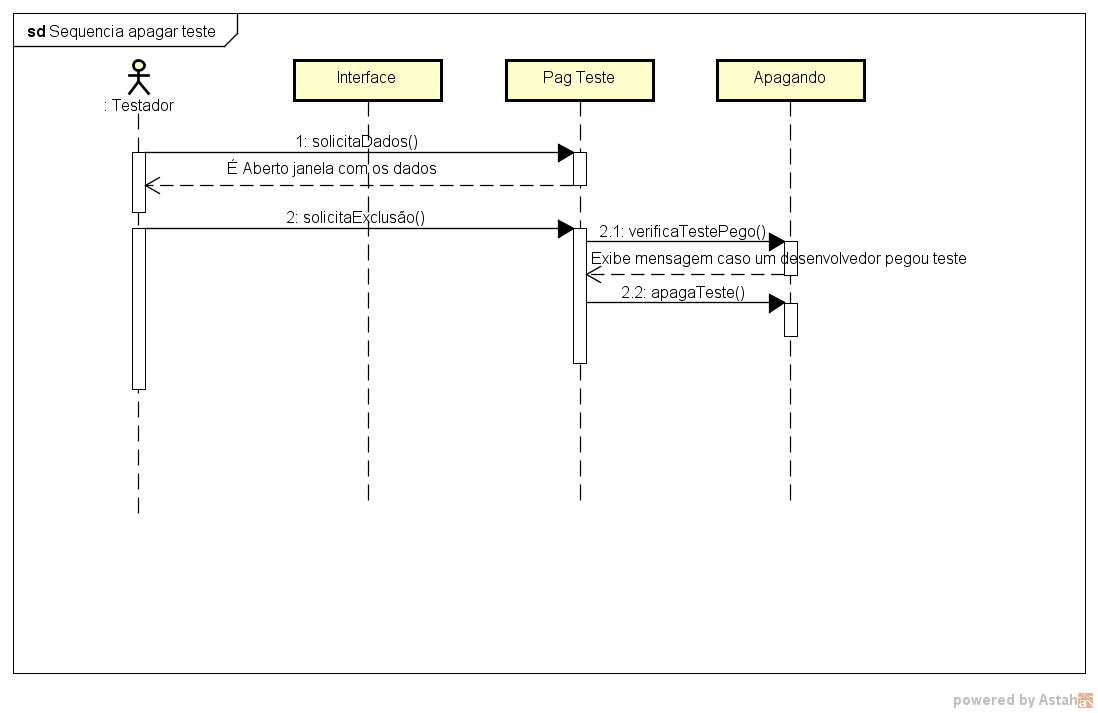


Figura 12: Diagrama de Sequência – Caso de uso 03

Fonte: Elaborada pelo autor

**5.4 UC04 Caso de Uso. Atualizar ocorrência**

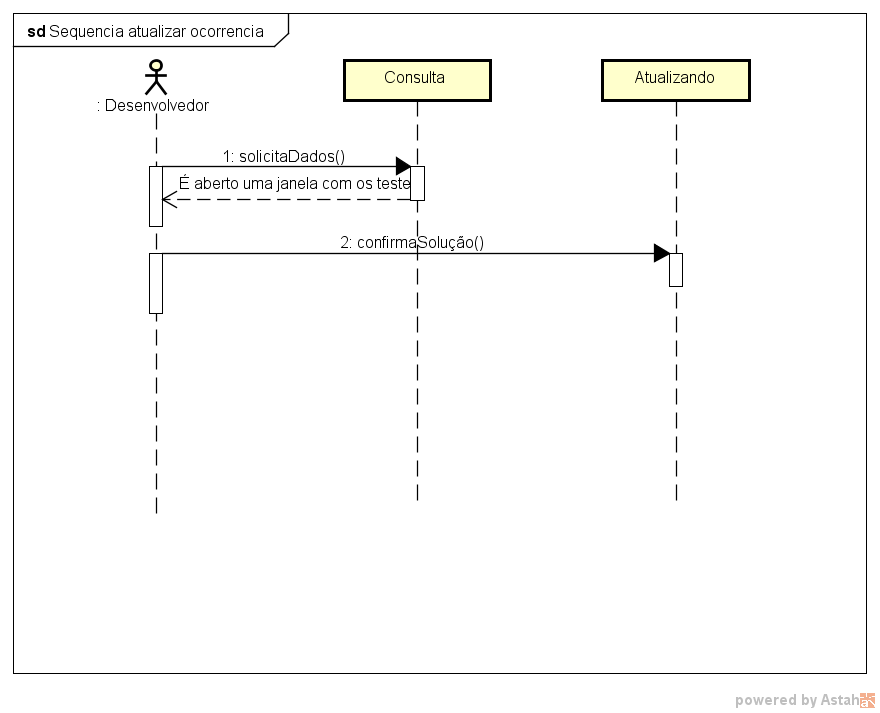


Figura 13: Diagrama de Sequência – Caso de uso 04

Fonte: Elaborada pelo autor

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

Nenhuma