Ticket Machine – Inspeção documentação

**\*Problema formulado pelo Professor Calebe de Paula Bianchini**

**Alunos:**

# 1 Engenharia de Requisitos

As estações de trem frequentemente fornecem máquinas de vender bilhetes que imprimem um bilhete quando um cliente insere a quantia correta para pagar a passagem. As máquinas mantêm uma soma total da quantidade de dinheiro que coletou durante toda sua operação.

## 1.1 Catálogo dos Atores

|  |  |
| --- | --- |
| **Ator** | **Descrição** |
| Cliente | Usuário da máquina de vender bilhetes. |

## 1.2 Diagrama dos Casos de Uso



## 1.3 Especificação dos Casos de Uso

### 1.3.1 CSU01 – Inserir dinheiro

|  |  |
| --- | --- |
| **Identificador** | CSU01 |
| **Nome** | Inserir dinheiro |
| **Atores** | Cliente |
| **Sumário** | Uma quantia de dinheiro é inserida na máquina. É importante ressaltar que essa quantia sempre é representada por uma única nota de papel-moeda. |
| **Complexidade** | Médio |
| **Regras de Negócio** | N/D |
| **Pré-condições** | N/D |
| **Pós-condição** | O valor inserido é adicionado ao saldo total disponível. |
| **Pontos de Inclusão** | N/D |
| **Pontos de Extensão** | N/D |

|  |  |
| --- | --- |
| **Fluxo Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O Cliente insere uma nota de papel-moeda. |  |
|  | 2. O Sistema mostra uma mensagem informando para o Cliente aguardar alguns instantes. |
|  | 3. O Sistema valida a nota de papel-moeda inserida. |
|  | 4. O Sistema adiciona o valor ao saldo disponível. |
|  | 5. O Sistema informa que a nota de papel-moeda foi aceita. |
|  | 6. O sistema informa o saldo atual. Caso de Uso é encerrado. |

|  |  |
| --- | --- |
| **Fluxo de Exceção 1: 3a. O Sistema não valida a nota de papel-moeda.** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. O Sistema devolve a nota de papel-moeda. |
|  | 2. Enquanto o Cliente não retirar a nota de papel-moeda, o Sistema informa que a nota de papel-moeda é inválida. |
| 3. O Cliente retira a nota de papel-moeda. |  |
|  | 4. Volta ao passo 6 do Fluxo Principal. |

### 1.3.2 CSU02 – Solicitar bilhete

|  |  |
| --- | --- |
| **Identificador** | CSU02 |
| **Nome** | Solicitar bilhete |
| **Atores** | Cliente |
| **Sumário** | Um único bilhete de transporte é impresso. |
| **Complexidade** | Fácil |
| **Regras de Negócio** | N/D |
| **Pré-condições** | N/D |
| **Pós-condição** | Um bilhete é impresso e seu valor é debitado do saldo disponível. |
| **Pontos de Inclusão** | N/D |
| **Pontos de Extensão** | N/D |

|  |  |
| --- | --- |
| **Fluxo Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O Cliente solicita impressão do bilhete de transporte. |  |
|  | 2. O Sistema verifica o saldo e emite o bilhete. |
|  | 3. O sistema informa o saldo atual. Caso de Uso é encerrado. |

|  |  |
| --- | --- |
| **Fluxo de Exceção 1: 2a. O saldo é insuficiente para emissão do bilhete.** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. O Sistema informa que o saldo é insuficiente. |
|  | 2. Volta ao passo 3 do Fluxo Principal. |

### 1.3.2 CSU03 – Solicitar troco

|  |  |
| --- | --- |
| **Identificador** | CSU03 |
| **Nome** | Solicitar troco |
| **Atores** | Cliente |
| **Sumário** | O troco é devolvido. Vale ressaltar que o troco também é em nota de papel-moeda. |
| **Complexidade** | Fácil |
| **Regras de Negócio** | N/D |
| **Pré-condições** | N/D |
| **Pós-condição** | O valor em nota de papel-moeda é devolvido e o saldo disponível é zerado. |
| **Pontos de Inclusão** | N/D |
| **Pontos de Extensão** | N/D |

|  |  |
| --- | --- |
| **Fluxo Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O Cliente solicita devolução do saldo restante como troco. |  |
|  | 2. O Sistema verifica o saldo e devolve a quantidade do saldo em notas de papel-moeda. |
|  | 3. O Sistema zera o valor do saldo. |
|  | 3. O sistema informa o saldo atual. Caso de Uso é encerrado. |

## 1.4 Protótipos

### 1.4.1 Tela principal



# 2 Projeto Orientado a Objetos

## 2.1 Diagrama de Classes Principal



## 2.2 Diagrama de Classes de interação



## 2.3 Diagrama de Sequência – Inserir dinheiro



## 2.4 Diagrama de Seqüência – Solicitar bilhete



## 2.5 Diagrama de Seqüência – Solicitar troco



É importante salientar que estes diagramas representam apenas o fluxo principal dos casos de uso.

# 3 Inspeção

Crie um relatório com o resultado da inspeção deste documento. Liste os defeitos e apresente onde foram encontrados. Classifique os defeitos por categoria (veja tabela abaixo) e quanto à severidade (baixa/média/alta).

Tabela 1. Defeitos de requisitos [Shull 1998].

|  |  |
| --- | --- |
| **Tipos de Defeito** | **Descrição** |
| Omissão | Deve-se à omissão ou negligência de alguma informação necessária ao desenvolvimento do software. |
| Ambiguidade | Ocorre quando uma determinada informação não é bem definida, permitindo assim uma interpretação subjetiva, que pode levar a múltiplas interpretações. |
| Fato incorreto | Informações dos artefatos do sistema que são contraditórias com o conhecimento que se tem do domínio da aplicação. |
| Inconsistência | Ocorre quando duas ou mais informações são contraditórias entre si. |
| Informação estranha | Informação desnecessária incluída nos requisitos do software que esta sendo desenvolvido |

Tabela 2. Defeitos de código [Jones 2009].

|  |  |
| --- | --- |
| **Tipos de Defeito** | **Descrição** |
| Comissão | Ocorre quando existe algum segmento de código que foi implementado incorretamente, i.e., cuja implementação é diferente do que foi especificado. |
| Inicialização | Ocorre quando se tenta acessar uma variável que não foi inicializada. |
| Computação | Similar ao defeito de comissão; ocorre quando um valor é definido erroneamente para uma variável. |
| Desempenho | Algumas rotinas executam comandos ou laços (*loops*) desnecessários. |
| Controle | Ocorre quando um comando de desvio condicional é usado de forma incorreta. |
| Excesso | Existem trechos de código irrelevantes e desnecessários. |
| Dados | Ocorre quando uma estrutura de dados é manipulada de forma incorreta (por exemplo, quando se tenta acessar um índice inexistente de um vetor/matriz). |

# 4 Referências

Pötter, H.; Schots, M. , 2011. InspectorX: *Um Jogo para o Aprendizado em Inspeção de Software*. Anais do FEES11 - Fórum de Educação em Engenharia de Software, São Paulo-SP, Brasil.

Jones, C., 2009. *Software Engineering Best Practices*, McGraw-Hill Inc., New York, USA.

Shull, F., 1998. *Developing Techniques for Using Software Documents: A Series of Empirical Studies*, Ph.D. Thesis, University of Maryland, College Park.