Engenharia de Software Orientada a Modelos

Frederico Farias Martins

30 de agosto de 2020

Sumário

Introdução

O presente relatório tem como finalidade comunicar informações organizadas a partir de um enunciado de programação. O foco deste trabalho é de utilizar os conceitos vistos em aula para a modelagem de uma solução para o enunciado. Para isso, foi escolhido o enunciado 1009 do site *URI Online Judge*[3].

1.1 Enunciado 1009 - Salário com bônus

O enunciado 1009 consiste em desenvolver um sistema capaz de ler o nome de um vendedor, o salário fixo e o total de vendas efetuadas no mês[1]. Além disso, o programa deve calcular o total que o vendedor deverá receber, uma vez que ele ganha 15% da comissão sobre suas vendas.

A figura abaixo representa um exemplos de entrada e saída conforme consta no enunciado[3]. Vale lembrar que a esquerda na figura estão os dados do vendedor: nome, salário fixo e total de vendas, respectivamente.

Exemplos de Entrada	Exemplos de Saída
JOAO 500.00 1230.30	TOTAL = R$ 684.54
PEDRO 700.00 0.00	TOTAL = R$ 700.00
MANGOJATA 1700.00 1230.50	TOTAL = R$ 1884.58

Figura 1.1: **Exemplos de entrada e saída do programa**

[1]O total de vendas deve ser representado pela quantidade em dinheiro

Modelo Inclusivo

O Modelo Inclusivo utilizado para o desenvolvimento deste trabalho consiste em um diagrama de casos de uso produzido na ferramenta Astah[2]. O diagrama pode ser conferido na figura 2.1.

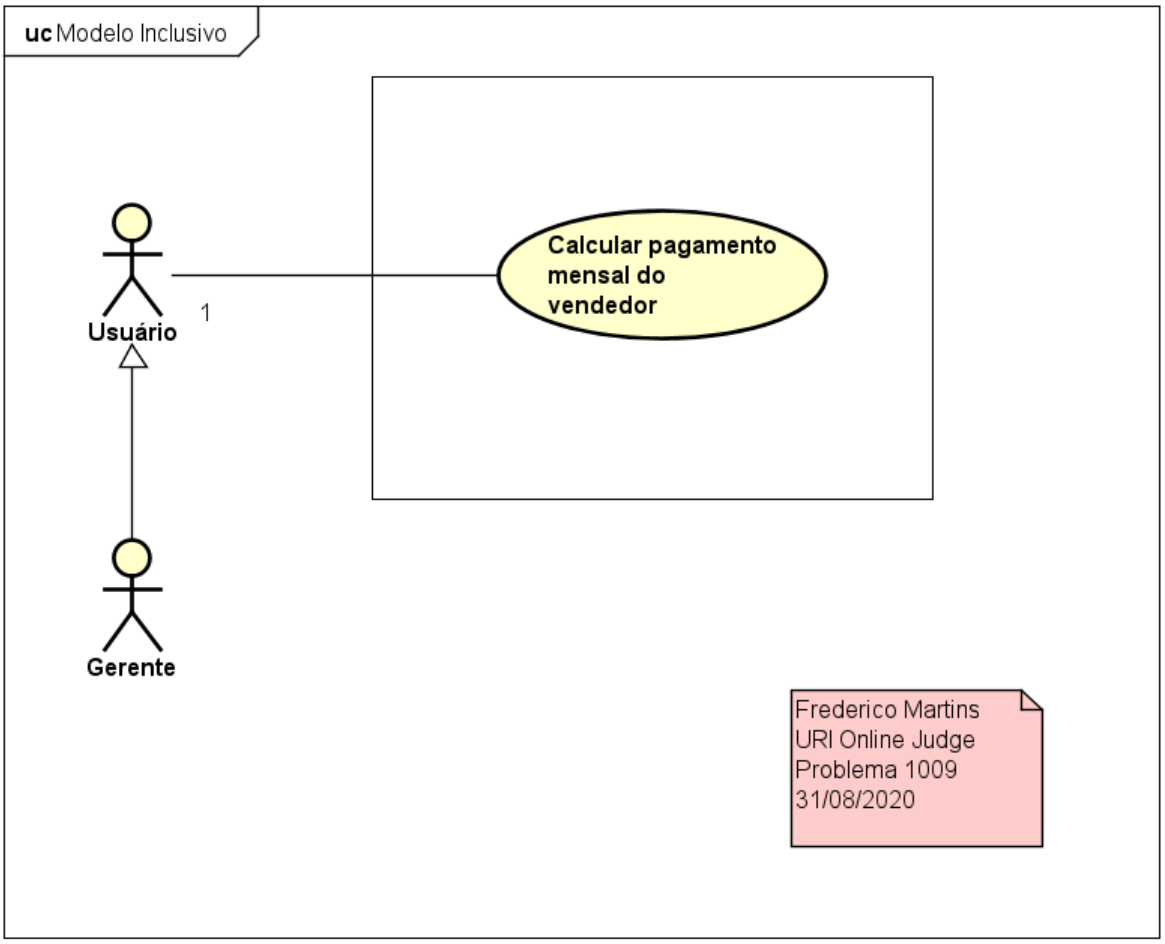

Figura 2.1: **Diagrama de casos de usos**

Com base na figura 2.1, é possível observar que o diagrama não apresenta alto nível de complexidade em virtude da simplicidade do enunciado proposto pelo URI Online Judge[3]. Além disso, é importante ressaltar a presença da descrição do caso de uso, responsável por dar mais detalhamento ao caso. A descrição pode ser vista na linha "*Base Sequence*"na figura 2.2.

Considerando a descrição apresentada na figura 2.2, é possível observar que um caso de uso é o suficiente para cobrir os requisitos presentes no enunciado 1009 do URI Online Judge.

ITEM	VALUE
UseCase	Calcular pagamento mensal do vendedor
Summary	
Actor	Usuário
Precondition	
Postcondition	
Base Sequence	Usuário informa o nome do Vendedor Usuário informa o salário fixo do vendedor Usuário informa o valor das vendas do mês Sistema calcula e apresenta o valor total_Mes = salarioFixo + (venda_Mes * 0.15)
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

Figura 2.2: **Descrição do Caso de Uso**

Modelo Independente de Plataforma

Para o desenvolvimento do Modelo Independente de Plataforma (também chamado de PIM) foi utilizados dois diagrama de classes UML. Em primeiro lugar, para garantir que estamos tratando de um modelo que não depende de plataforma de programação, foi desenvolvido o diagrama *Primitive Types*, que pode ser observado na figura 3.1.

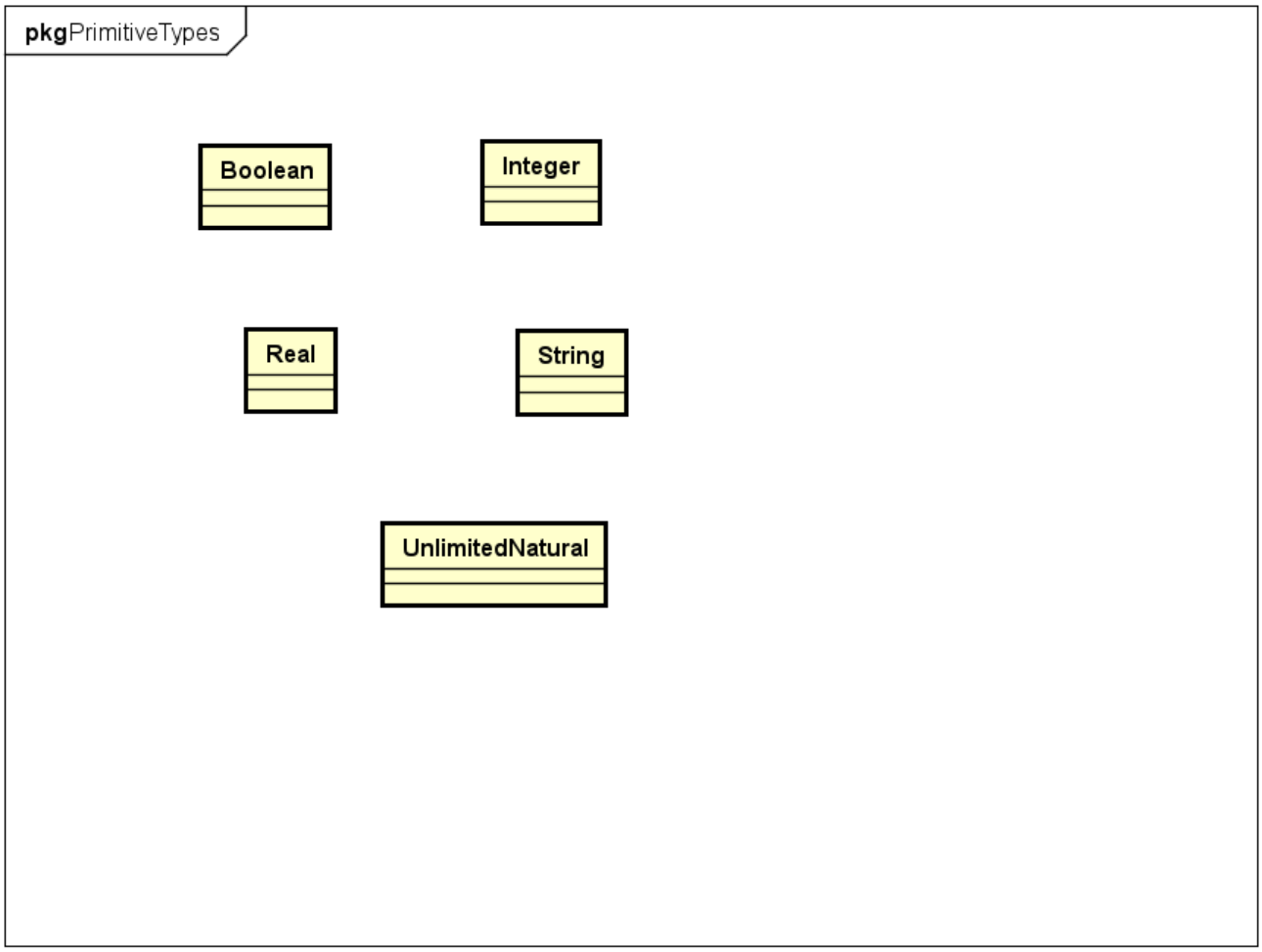

Figura 3.1: **Primitive Types**

É importante destacar que, embora o arquivo XMI esteja disponível na página de especificações do UML[1], foi necessário fazer o diagrama *Primitive Types* a mão em virtude de problemas técnicos com o plugin "*XMI Import*"do Astah.

O PIM, por sua vez, corresponde a um diagrama de classe bastante simples. Trata-se de uma classe responsável por cobrir os requisitos propostos pela descrição do caso de uso:

- Receber o nome do vendedor

- Receber o salário fixo do vendedor
- Receber o número de vendas no mês (em dinheiro)
- Retornar o total a pagar para o respectivo vendedor

O diagrama pode ser conferido na figura 3.2.

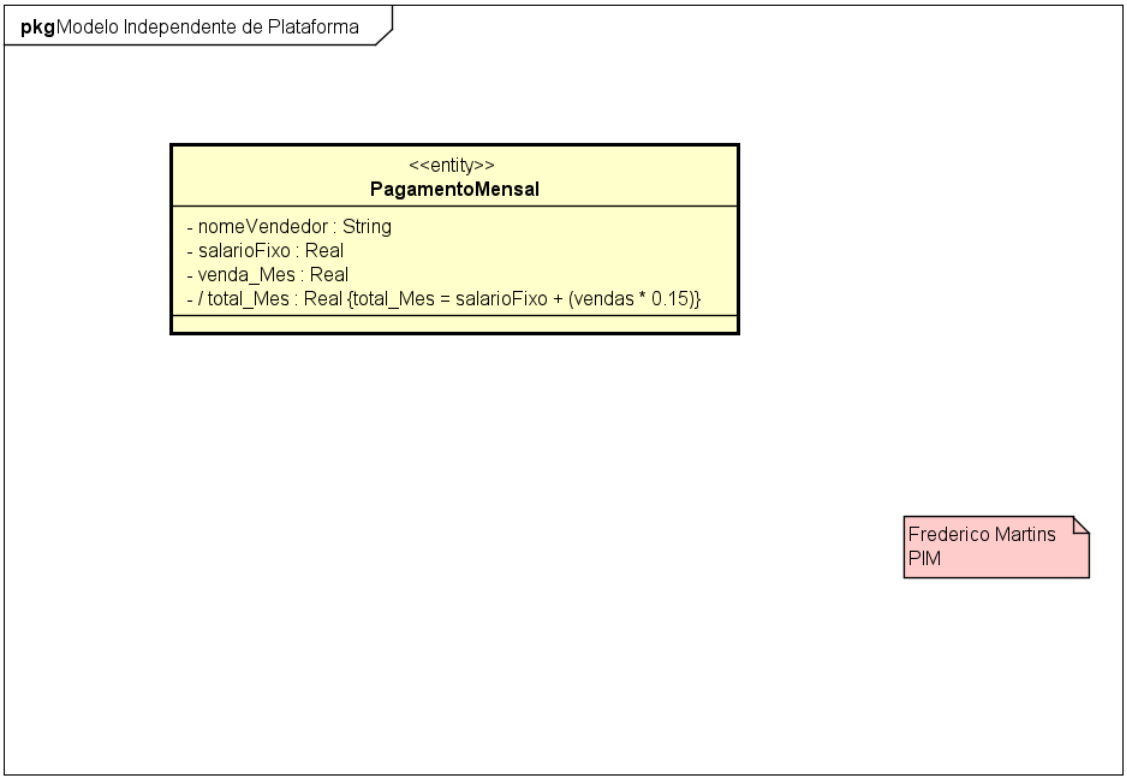

Figura 3.2: **PIM**

A figura 3.2 Mostra o diagrama de classe proposto para o enunciado 1009. É importante observar a variável "total_Mes"já é calculada pelo sistema. Conforme apresentado pelo enunciado 1009, o objetivo da variável é retornar o valor do total a ser pago para o vendedor informado pelo usuário.

Lista de verificação

Foi desenvolvida uma lista de verificação para cada um dos modelos. A figura 4.1 apresenta duas tabelas simples desenvolvidas no Excel para representar as listas de verificação.

Modelo Inclusivo	
Item para verificação	**Situação**
Casos de uso cobrem os requisitos do enunciado	OK
Os casos de uso possuem decrições adequadas	OK
É explicitado que apenas um usuário deve operar o programa	OK

PIM	
Item para verificação	**Situação**
As classes possuem nomes adequados	OK
Os atributos das classes possuem nomes claros e adequados	OK
A classe faz o cálculo do salário conforme descrito no enunciado	OK

Figura 4.1: **Lista de verificação para os modelos**

Vale ressaltar que as listas têm baixa complexidade em virtude da simplicidade do problema. Foi decidido pelo autor do presente trabalho que uma lista simples com a da figura 4.1 cobre o necessário para o desenvolvimento correto do software que resolve o enunciado 1009.

Repositório

O presente trabalho está disponível no **GitHub**. Para acessá-lo, basta clicar no link informado abaixo:

`https://github.com/FredericoMartins0/Revisao_UML`

Também é possível baixar o repositório para um diretório local. Para isso, é necessário inserir, por meio do terminal, o comando *git clone* seguido do endereço:

`https://github.com/FredericoMartins0/Revisao_UML.git`.

Referências Bibliográficas

[1] About the unified modeling language specification version 2.5.1. https://www.omg.org/spec/UML. Acessado em: 30/08/2020.

[2] The power of software modeling. https://astah.net/. Acessado em: 30/08/2020.

[3] Salário com Bônus. https://www.urionlinejudge.com.br/judge/pt/problems/view/1009. Acessado em: 27/08/2020.