**Especificação do Projeto**

**BPM Game Engine**

**Versão 1.0**

Fabio Takeshi Ishikawa

Janeiro de 2020

Contents

[**1. Introdução** 4](#_Toc34863285)

[1.1 O que é BPM Game Engine 4](#_Toc34863286)

[1.2 Motivação do Projeto 4](#_Toc34863287)

[1.3 Objetivo deste documento 4](#_Toc34863288)

[1.4 Público-alvo deste documento 4](#_Toc34863289)

[**2. Descrição Geral de BPM Game Engine** 5](#_Toc34863290)

[2.1 Funcionalidades essenciais 5](#_Toc34863291)

[2.2 Os Usuários 5](#_Toc34863292)

[2.3 Plataformas e Ferramentas 5](#_Toc34863293)

[2.4 Documentações 5](#_Toc34863294)

[2.5 Dependências 5](#_Toc34863295)

[**3. Processos de Software** 6](#_Toc34863296)

[3.1 Processo: Incremental Adaptado 6](#_Toc34863297)

[3.2 Descrição das Etapas do Projeto 6](#_Toc34863298)

[3.3 Descrição da análise e planejamento de cada iteração 6](#_Toc34863299)

[3.4 Descrição da execução da iteração 6](#_Toc34863300)

[**4. Engenharia de Requisitos** 7](#_Toc34863301)

[4.1 Como descrever requisitos de usuário e requisitos de sistema 7](#_Toc34863302)

[4.2 Como definir as prioridades e riscos dos requisitos 7](#_Toc34863303)

[4.3 Processo de elicitação e análise de requisitos 7](#_Toc34863304)

[**5. Modelagem Conceitual (é necessário?)** 8](#_Toc34863305)

[5.1 Cenários ou Diagramas UML? 8](#_Toc34863306)

[**6. Projeto de Software** 9](#_Toc34863307)

[6.1 Visão Geral de todas as ferramentas 9](#_Toc34863308)

[6.2 Arquitetura Game Engine (Core) 9](#_Toc34863309)

[6.3 Parser BPMN (Camunda) 9](#_Toc34863310)

[6.4 Método PYP – Play Your Process 9](#_Toc34863311)

[**7. Teste de Software** 10](#_Toc34863312)

[7.1 Plano de Testes 10](#_Toc34863313)

[7.2 Testes de Unidade Automatizadas 10](#_Toc34863314)

[7.3 Testes de Integração Automatizada 10](#_Toc34863315)

[7.4 Testes de Aceitação 10](#_Toc34863316)

[7.5 Documentação de Testes 10](#_Toc34863317)

[**8. Gerência de Configuração de Software** 11](#_Toc34863318)

[8.1 Slack – Comunicação entre a equipe 11](#_Toc34863319)

[8.2 Trello – Gestão de tarefas (Kanban) 11](#_Toc34863320)

[8.3 Git e Github – Controle de Versão 11](#_Toc34863321)

[8.3.1 Versionamento Semântico 11](#_Toc34863322)

[8.3.2 Workflow: Pull Request Workflow 11](#_Toc34863323)

[8.3.3 Workflow: No Switch Yard 11](#_Toc34863324)

[8.3.4 Guia de contribuição open source 11](#_Toc34863325)

[8.4 DevOps com Github, TravisCI e Codacy 11](#_Toc34863326)

[**9. Licenças** 12](#_Toc34863327)

[**10. Garantia de Qualidade** 13](#_Toc34863328)

# **1. Introdução**

Este documento tem como objetivo descrever em detalhes a especificação do projeto BPM Game Engine.

## 1.1 O que é BPM Game Engine

BPM Game Engine é um motor de jogo multiplataforma para criação de jogos de baseados em modelos de negócio1.

**Nota:** Jogos digitais com propósito que apresentam um processo de negócio de forma ludificada e permitem assim, aos jogadores, compreender e aprender seu funcionamento de forma divertida e engajante e desenvolver reflexões em relação à sua necessidade, sua prática, seus valores, seus desafios e limitações de execução. (CLASSE e ARAUJO).

## 1.2 Motivação do Projeto

## 1.3 Objetivo deste documento

Esta especificação pode ser utilizada como um “guia” para o desenvolvimento e uma forma de comunicação entre os envolvidos e visualização do estado atual do projeto.

## 1.4 Público-alvo deste documento

Este documento está destinado a todas as pessoas envolvidas no projeto e desenvolvimento de BPM Game Engine.

# **2. Descrição Geral de BPM Game Engine**

Nesta seção, estão descritas todas as funcionalidades que compõe BPM Game Engine.

## 2.1 Funcionalidades essenciais

## 2.2 Os Usuários

## 2.3 Plataformas e Ferramentas

## 2.4 Documentações

## 2.5 Dependências

# **3. Processos de Software**

Com o intuito de entregar resultados de alta qualidade e dentro do prazo estabelecido, estabelecemos uma série de etapas para manter o controle, organização e estabilidade do projeto. Estas etapas devem ser executadas de forma cíclica, onde cada ciclo representa uma entrega de valor. Esta seção descreve todas etapas e a análise, planejamento e execução de cada ciclo.

## 3.1 Processo: Incremental Adaptado

## 3.2 Descrição das Etapas do Projeto

* Análise do projeto (geral)
* Análise dos requisitos de software (geral)
* Plano de Desenvolvimento
* Plano de Teste
* Desenvolvimento
* Teste de Unidade
* Teste de Integração
* Teste de Aceitação
* Final da iteração – análise da iteração realizada e atualizações de todos os documentos

**Nota:** Os itens com fundo azul representam as etapas que utilização ferramentas automatizadas relacionada a DevOps (Github, Travis.CI, ReaderTheDocs, etc).

## 3.3 Descrição da análise e planejamento de cada iteração

## 3.4 Descrição da execução da iteração

# **4. Engenharia de Requisitos**

## 4.1 Como descrever requisitos de usuário e requisitos de sistema

## 4.2 Como definir as prioridades e riscos dos requisitos

## 4.3 Processo de elicitação e análise de requisitos

* Descoberta e compreensão dos requisitos
* Classificação e organização dos requisitos
* Priorização e negociação dos requisitos
* Documentação dos requisitos

# **5. Modelagem Conceitual (é necessário?)**

## 5.1 Cenários ou Diagramas UML?

# **6. Projeto de Software**

Nesta seção abordamos todos os componentes que fazem parte do software, assim como todos os componentes utilizados para construí-lo. É descrita também a arquitetura de BPM Game Engine e o mapeamento utilizado no método Play Your Process.

## 

## 6.1 Arquitetura Game Engine (Core)

## 6.2 Visão Geral de todas as ferramentas

### 6.2.1 Compiladores

Como a ideia do projeto é atender diversas plataformas, o projeto deve ser possível ser compilado por diversos compiladores. A seguir segue a lista de compiladores utilizados para compilar o projeto:

* GCC 9.2.0 (MinGW) – MinGW.org GCC Build-20200227-1

## 6.3 Parser BPMN (Camunda)

## 6.4 Método PYP – Play Your Process

# **7. Teste de Software**

## 7.1 Plano de Testes

## 7.2 Testes de Unidade Automatizadas

## 7.3 Testes de Integração Automatizada

## 7.4 Testes de Aceitação

## 7.5 Documentação de Testes

# **8. Gerência de Configuração de Software**

## 8.1 Slack – Comunicação entre a equipe

## 8.2 Trello – Gestão de tarefas (Kanban)

## 8.3 Git e Github – Controle de Versão

O sistema de controle utilizado neste projeto é o Git (<https://git-scm.com/>). A escolha de Git para o projeto é motivada pela sua popularidade e facilidade de integração com diversos projetos existentes. Além disso, ela é uma ferramenta livre e gratuita.

Para este projeto é recomendável utilizar qualquer versão a partir de **2.25.1**.



*Git Logo by Jason Long is licensed under the Creative Commons Attribution 3.0 Unported License.*

O projeto também com a plataforma online Github (<https://github.com/>) para a hospedagem do repositório do projeto. Esta plataforma também permite a integração com diversas ferramentas úteis e sistemas como o Travis-CI para automatização de builds e testes – práticas essências de integração contínua (DevOps).



## 8.3.1 Versionamento Semântico

## 8.3.2 Workflow: Pull Request Workflow

## 8.3.3 Workflow: No Switch Yard

## 8.3.4 Guia de contribuição open source

## 8.4 DevOps com Github, TravisCI e Codacy

# **9. Licenças**

# **10. Garantia de Qualidade**