Documentação de Definição de Requisitos

Projeto: Mar&moto
Responsável: Hanna Tátila de Sousa e Neimar Paulo Alves

Registro de alterações

Versão	Responsável	Data	Alterações
1.0	Hanna e Neimar	24/09/2015	
2.0	Hanna e Neimar	28/11/2015	-Inclusão dos Requisitos Funcionais: RF04 ao RF06; -Inclusão dos Requisitos Não Funcionais: RNF04 ao RNF11; -Melhora na explicação do Processo de Software utilizado

1. Introdução

Este documento apresenta os requisitos de usuário do Jogo Mar&moto e está organizado da seguinte forma: a Seção 2 contém uma descrição do propósito do sistema; a Seção 3 apresenta uma descrição do minimundo apresentando o problema; a Seção 4 apresenta as listas de requisitos de usuário; a seção 5 apresenta o Diagrama de Classe; a Seção 6 apresenta o Diagrama de Caso de Uso; na Seção 7, é apresentado o processo de software escolhido; a Seção 8 contém o cronograma.

2. Descrição do Propósito do Sistema

O sistema se trata de um jogo de distância que contém vários desafios e convida o usuário a percorrer longas distâncias, marcando muitos pontos. O jogo é sem fim, não possui níveis. A única finalidade é percorrer o maior número de distância possível, não ser atingido pelos obstáculos e coletar os bônus disponíveis na fase.

3. Descrição do Minimundo

A empresa desenvolvedora de jogos HTSGames precisa criar um novo jogo que prenda a atenção e desafie seus jogadores. Pensando nisso, a HTSGames planeja criar um jogo de distância.

Esse jogo possui como personagem principal um adorável peixinho montado em sua moto radical. O peixinho precisa desviar de vários obstáculos marinhos que aparecem em seu caminho. A cada obstáculo ultrapassado, a pontuação é incrementada. Sendo que o personagem possui três vidas iniciais. A cada vez que é atingido por um obstáculo, o personagem perde uma vida. Se todas as vidas forem perdidas, o jogo é finalizado. Ao decorrer do jogo, aparecerão bônus que podem ser coletados e transformados em habilidades ou uma nova vida. Essas habilidades podem ser: escudo ou estrelinha.

Além disso, ao iniciar o jogo, o jogador precisa informar um nome que depois será associado à pontuação máxima atingida pelo jogador. O cadastro de nome possui algumas restrições, como: quantidade máxima de 10 caracteres, sendo que não pode conter caracteres especiais. Toda vez ao entrar no jogo, o jogador deverá inserir seu nome. Se o nome apresentado já estiver cadastrado, então sua pontuação máxima será atualizada.

O jogador terá a possibilidade de consultar o ranking do jogo, que exibe, ordenadamente, a pontuação de cada jogador.

4. Requisitos de Usuário

Tomando por base o contexto do sistema, foram identificados os seguintes requisitos de usuário:

Requisitos Funcionais

Identificador	Descrição	Prioridade	Depende de
RF01	O sistema deve permitir o jogador iniciar um jogo	Alta	
RF02	O sistema deve cadastrar login e	Alta	

	senha dos usuários.		
RF03	O sistema deve registrar e atualizar a pontuação máxima alcançada por cada jogador.	Alta	RF02
RF03	O sistema deve gerar um ranking que exibe ordenadamente a pontuação máxima de todos os jogadores.	Alta	RF02, RF03
RF04	O sistema deve permitir o jogador desistir no meio da partida	Média	RF01
RF05	O sistema deve permitir o jogador jogar novamente	Média	RF01
RF06	O sistema deve permitir o usuário encerrar a partida	Alta	RF01

Requisitos Não Funcionais

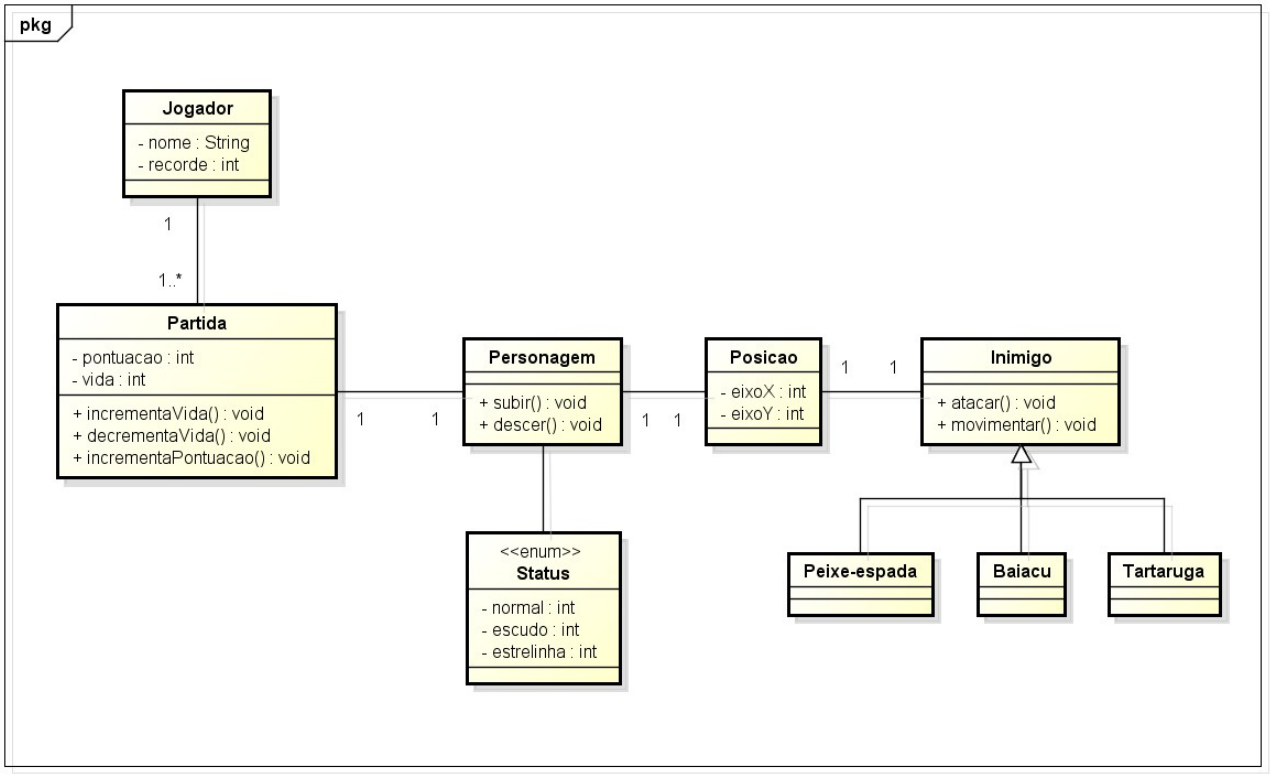
Identificador	Descrição	Prioridade	Depende de
RNF01	O jogo dever ser desenvolvido para computadores	Alta	
RNF02	A persistência das informações deve ser implementada, em um primeiro momento, em um Sistema Gerenciador de Bancos de Dados Relacionais (SGBDR) livre (Postgres ou MySQL).	Alta	
RNF03	O jogo deve ser desenvolvido em Python 2.7 usando framework Pygame 1.9.2	Alta	

RNF04	O jogo dever ser controlado pelo teclado	Alta	
RNF05	O jogo será offline	Alta	
RNF06	O software deve ter controle de versão no github	Alta	
RNF07	Todos os dados de entrada devem ser validados pelo jogo	Alta	
RNF08	O jogo deve ser de fácil manutenção	Alta	
RNF09	O ranking do jogo estará disponível em um servidor web	Média	
RNF10	O jogo poderá sofrer mudanças de banco de dados	Média	
RNF11	Todos os módulos devem ser testados individualmente, tanto o comportamento normal e excepcional de cada função dos módulos	Alta	

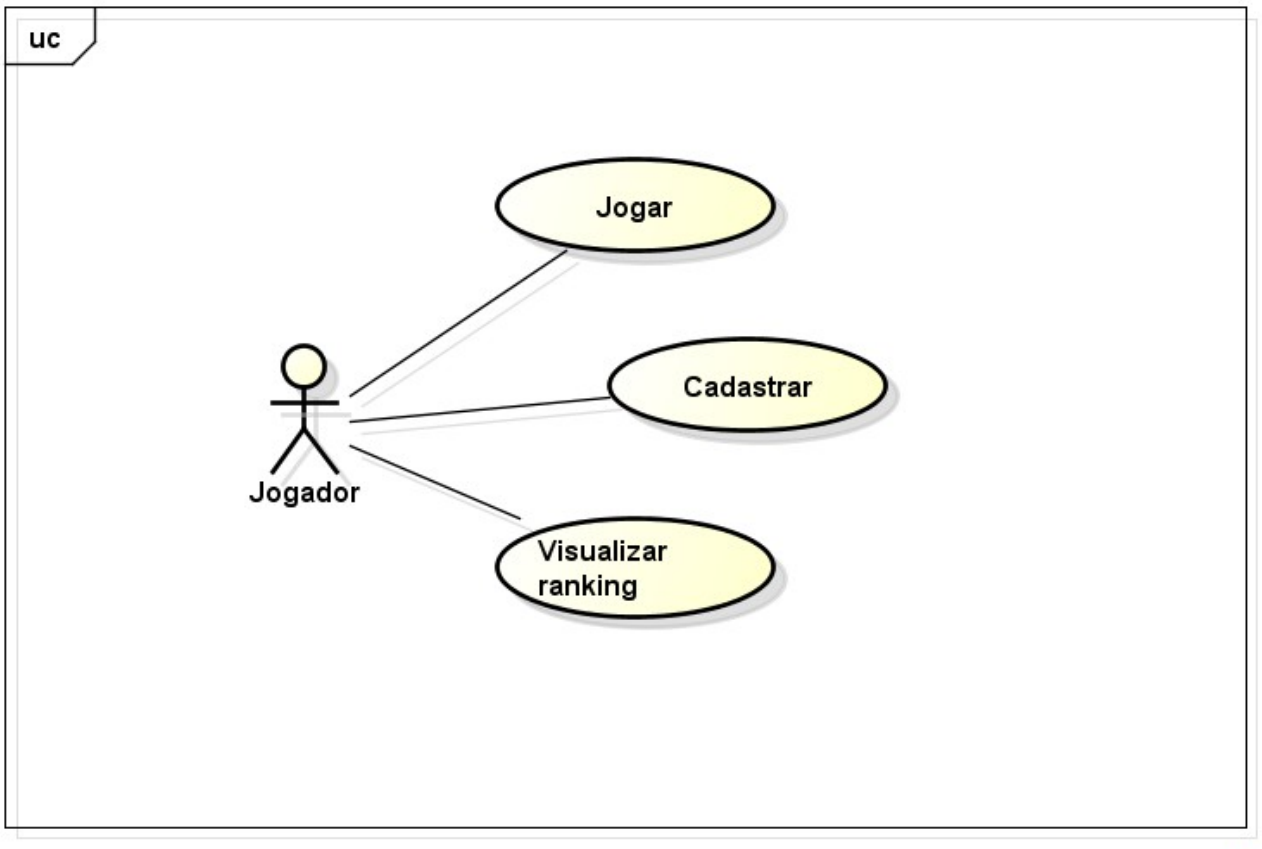
Regras de Negócio

Identificador	Descrição	Prioridade	Depende de
RN01	O sistema deve validar a inserção de nicks, não permitindo que possuam mais de 10 caracteres, assim como caracteres especiais.	Alta	
RN02	O jogo deve ser finalizado assim que o jogador perder todas as vidas.	Alta	

5. Diagrama de Classe



6. Diagrama de Caso de Uso



7. Processo de Software

Para implementação do sistema, escolhemos fazer uso de Metodologia Ágil. As metodologias ágeis têm o objetivo de acelerar o desenvolvimento do software visando a melhoria contínua do processo, gerando benefícios como o aumento da comunicação e interação da equipe, organização diária para o alcance da meta definida, evitar falhas na elaboração, respostas rápidas às mudanças e aumento significativo da produtividade.

A HTSGames utilizará o processo Scrum de Metodologia Ágil. O Scrum é um processo de desenvolvimento iterativo e incremental para gerenciamento de projetos e desenvolvimento ágil de software. É utilizado para trabalhos complexos nos quais é impossível predizer tudo o que ocorrerá.

No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. O Sprint representa um tempo definido dentro do qual um conjunto de atividades deve ser executado. Metodologias ágeis de desenvolvimento de software são iterativas, ou seja, o trabalho é dividido em iterações, que no Scrum são chamadas de Sprints e geralmente duram de 2 a 4 semanas.

As funcionalidades a serem implementadas no projeto são mantidas em uma lista que é conhecida como Product Backlog. No início de cada Sprint, faz-se um Sprint Planning Meeting (uma reunião de planejamento), na qual o Product Owner (quem representa os envolvidos) prioriza todos os itens do Product Backlog e a equipe seleciona as funcionalidades que ela será capaz de implementar durante o Sprint que se inicia. As funcionalidades alocadas em um Sprint são transferidas do Product Backlog para o Sprint Backlog.

Ao final de um Sprint, a equipe apresenta as funcionalidades implementadas em uma Sprint Review Meeting onde o time mostra o que foi alcançado neste Sprint. Finalmente, faz-se uma Sprint Retrospective para identificar o que funcionou bem e o que pode ser melhorado e a equipe inicia o planejamento do próximo Sprint.

A Figura 1 representa um esquema de todo o funcionamento do Scrum.

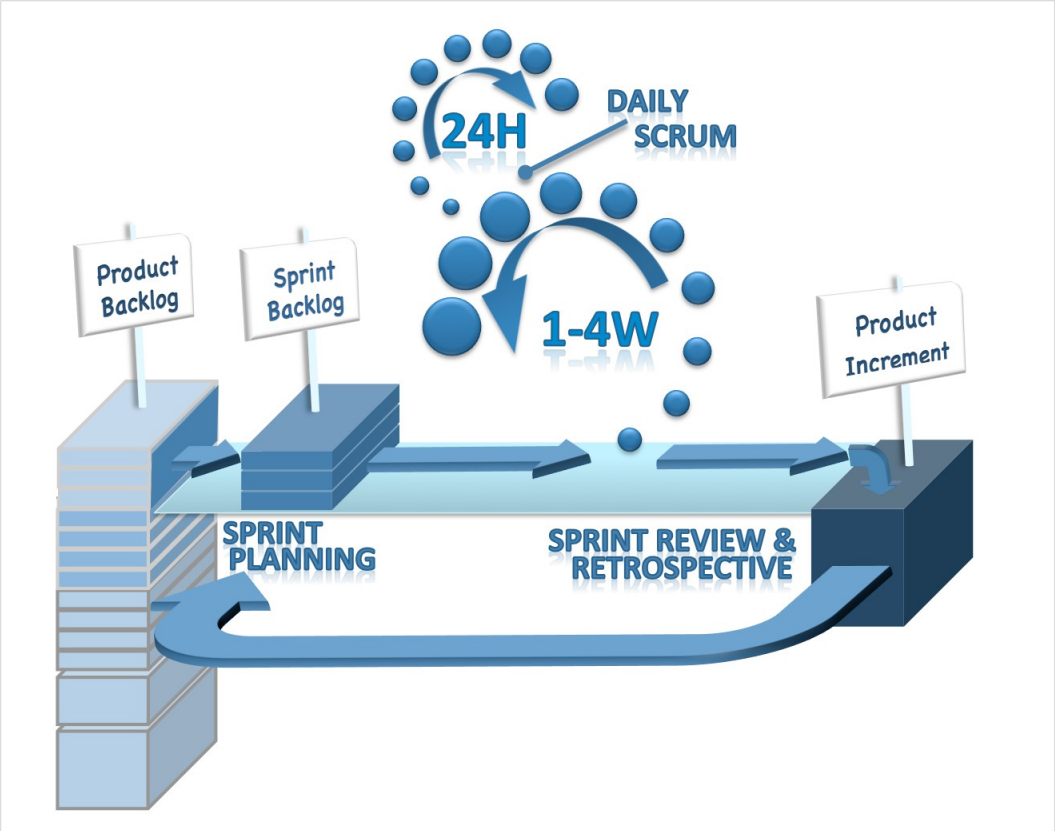


Figura 1: Esquema de um processo Scrum

Achamos que essa metodologia se adequaria à realidade do nosso sistema porque, por se tratar de um jogo, seria possível implementarmos algumas funcionalidades do jogo, obter um feedback do cliente e logo depois incrementar o jogo adicionando novas funcionalidades ou ajustando as já implementadas, tendo assim uma resposta rápida às mudanças. E como já percebemos, um jogo pode sofrer diversas mudanças durante sua implementação. Teríamos uma melhor organização para alcançar a meta definida a cada Sprint. Devido à boa comunicação e organização entre a equipe e os clientes, evitaríamos falhas na elaboração.

Sendo assim, seguiremos todos as etapas descritas de um processo Scrum.

8. Cronograma

	Análise	Projeto	Implementação	Testes	Implantação
Setembro	x				
Outubro		x			
Novembro			x		
Dezembro			x	x	
Janeiro				x	x