# Escopo do Projeto e do Produto - Monopoly

## Introdução

### Propósito do Documento

Este documento estabelece o escopo completo para o projeto da disciplina Gerência de Projeto e Manutenção de Software. Seu objetivo é criar uma base de entendimento comum entre todos os stakeholders sobre os objetivos, entregáveis, requisitos, premissas e restrições do projeto. Ele servirá como guia para o planejamento, execução e controle de todas as atividades do projeto, constituindo a *baseline* de escopo.

### Visão Geral do Projeto

O projeto consiste no desenvolvimento de uma versão digital do clássico jogo de tabuleiro Monopoly. O projeto será executado no contexto da disciplina, funcionando como um estudo de caso prático para a aplicação de conceitos de planejamento, estimativa, monitoramento, controle de configuração e manutenção de software. A equipe atuará como uma pequena *software house*, focando tanto na qualidade do produto final quanto na disciplina do processo de desenvolvimento.

## Escopo do Projeto

### Objetivos do Projeto

Os seguintes objetivos devem ser alcançados ao final do projeto:

* **OBJ-PROJ-01:** Aplicar um processo de desenvolvimento de software iterativo e incremental para gerenciar o ciclo de vida do produto.
* **OBJ-PROJ-02:** Planejar e documentar formalmente o escopo, cronograma, custos e riscos do projeto.
* **OBJ-PROJ-03:** Utilizar sistematicamente um Sistema de Controle de Versões (Git) para gerenciar todos os artefatos do projeto (código-fonte, documentação, apresentações).
* **OBJ-PROJ-04:** Monitorar e controlar o progresso do projeto comparando o planejado versus o realizado, utilizando técnicas como Gráfico de Burndown e Análise de Valor Agregado (EVA).
* **OBJ-PROJ-05:** Entregar um software funcional que implemente as regras fundamentais do jogo Monopoly, permitindo uma partida completa do início ao fim.

### Entregáveis do Projeto

#### Documentação de Gerenciamento

* Documento de Definição de Escopo.
* Estrutura Analítica do Projeto (EAP).
* Planilha de Estimativas de Esforço e Custo.
* Cronograma do Projeto (Gráfico de Gantt).
* Plano de Gerenciamento de Riscos.
* Relatórios de Monitoramento (Burndown, EVA).

#### Software

* Código-fonte completo e comentado do jogo.
* Versões executáveis do produto entregues em cada marco de apresentação.

#### Apresentações

* Conjunto de slides para cada uma das três apresentações de status do projeto.

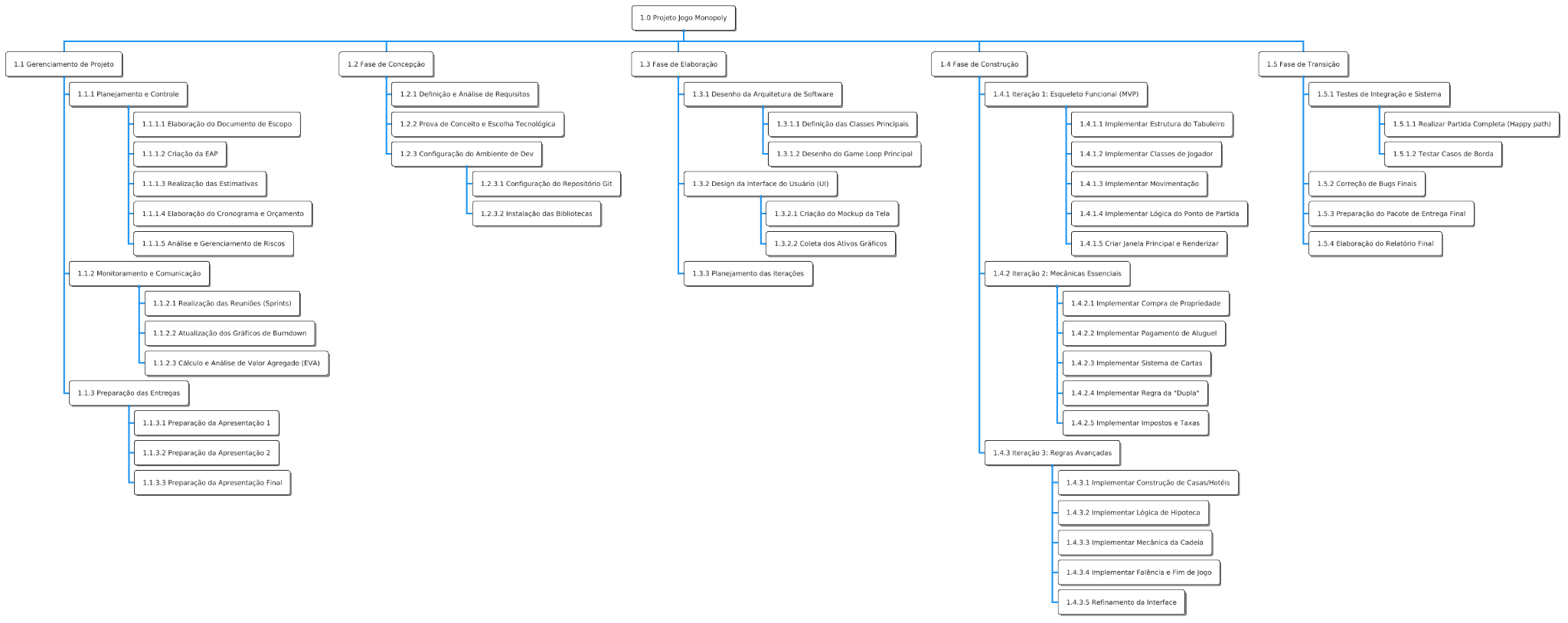
#### Repositório

* Repositório Git configurado contendo todos os artefatos mencionados acima.

### Estrutura Analítica do Projeto (EAP)

A Estrutura Analítica do Projeto (EAP), também conhecida como Work Breakdown Structure (WBS), é uma decomposição hierárquica de todo o trabalho a ser executado pela equipe para atingir os objetivos do projeto e criar os entregáveis requeridos. Ela organiza e define o escopo total do projeto, dividindo-o em pacotes de trabalho menores e mais gerenciáveis.

A EAP a seguir foi estruturada utilizando uma abordagem híbrida: os níveis superiores representam as fases do Processo Unificado, enquanto os níveis inferiores, dentro da fase de Construção, são orientados aos entregáveis de software definidos no escopo do produto.

****

*Figura 1 - Estrutura Analítica do Projeto (EAP) do Jogo Monopoly.*

## Escopo do Produto

O escopo do produto define as funcionalidades do jogo e foi derivado exclusivamente do manual de regras do Monopoly.

### Exclusões do Projeto (Fora do Escopo)

As seguintes funcionalidades e características estão explicitamente fora do escopo desta versão do projeto:

* Funcionalidade de multiplayer online (o foco será em multiplayer local ou jogador vs. IA).
* Gráficos 3D avançados ou animações complexas.
* Compatibilidade com múltiplos sistemas operacionais (será definida uma única plataforma-alvo, Windows Desktop ou Navegador Web Moderno).
* Implementação de "regras da casa" ou variações não descritas no manual oficial fornecido.
* Criação de contas de usuário, login ou persistência de histórico de partidas entre sessões.
* Suporte a múltiplos idiomas.

### Escopo do Produto (Requisitos Funcionais)

#### Setup e Configuração Inicial do Jogo

* **REQ-FUNC-1.1:** O sistema deve permitir que de 2 a 8 jogadores iniciem uma partida, escolhendo uma das peças disponíveis.
* **REQ-FUNC-1.2:** O sistema deve permitir a designação de um jogador como "Banqueiro".
* **REQ-FUNC-1.3:** O sistema deve distribuir automaticamente o capital inicial de $1500 para cada jogador, conforme a distribuição de notas especificada.
* **REQ-FUNC-1.4:** O sistema deve preparar o tabuleiro, posicionando as cartas "Sorte" e "Cofre" embaralhadas e as peças dos jogadores no "Ponto de Partida".

#### Mecânicas Principais do Jogo

* **REQ-FUNC-2.1:** O sistema deve gerenciar os turnos dos jogadores, começando pelo que obtiver o maior resultado no lançamento de dois dados.
* **REQ-FUNC-2.2:** Em cada turno, o sistema deve permitir ao jogador lançar os dados e mover sua peça automaticamente.
* **REQ-FUNC-2.3:** O sistema deve implementar a regra da "dupla": jogar novamente ao tirar números iguais e ir para a cadeia na terceira dupla consecutiva.
* **REQ-FUNC-2.4:** O sistema deve pagar o salário de $200 a um jogador sempre que ele parar ou passar pelo "Ponto de Partida".
* **REQ-FUNC-2.5:** O sistema deve apresentar e executar as instruções das cartas "Sorte" e "Cofre" quando um jogador parar na casa correspondente.

#### Gerenciamento de Propriedades

* **REQ-FUNC-3.1:** O sistema deve oferecer a um jogador a opção de comprar uma propriedade sem dono na qual ele tenha parado.
* **REQ-FUNC-3.2:** O sistema deve iniciar um leilão para todos os jogadores caso a opção de compra direta seja recusada.
* **REQ-FUNC-3.3:** O sistema deve calcular e cobrar automaticamente o aluguel quando um jogador parar em uma propriedade de outro.
* **REQ-FUNC-3.4:** O sistema deve permitir a venda de propriedades (sem construções) entre jogadores por um valor mutuamente acordado.

#### Construção e Melhorias

* **REQ-FUNC-4.1:** O sistema deve permitir que um jogador que possua um Monopólio (todas as propriedades de uma cor) compre casas e hotéis.
* **REQ-FUNC-4.2:** O sistema deve aplicar a regra de construção uniforme (não é possível construir uma segunda casa antes que todas as propriedades do grupo tenham uma).
* **REQ-FUNC-4.3:** O sistema deve permitir a venda de construções de volta ao Banco pela metade do preço, respeitando a regra de venda uniforme.

#### Transações e Condições Especiais

* **REQ-FUNC-5.1:** O sistema deve gerenciar o pagamento de impostos e taxas.
* **REQ-FUNC-5.2:** O sistema deve permitir que jogadores hipotequem propriedades ao Banco e as resgatem posteriormente com juros de 10%.
* **REQ-FUNC-5.3:** O sistema deve implementar todas as regras da "Cadeia", incluindo as condições de entrada e as três formas de saída (pagar fiança, usar carta, tirar dupla).
* **REQ-FUNC-5.4:** O sistema deve detectar a condição de falência de um jogador e gerenciar a transferência de seus ativos para o credor (outro jogador ou o Banco).
* **REQ-FUNC-5.5:** O sistema deve declarar um vencedor quando todos os outros jogadores forem à falência.

## Stakeholders

| **Stakeholder** | **Papel / Interesse** |
| --- | --- |
| Troy Costa Kohwalter | Cliente, Avaliador. Define os requisitos do projeto e avalia os entregáveis. |
| Equipe de Desenvolvimento | "Software House". Responsável por planejar, executar e entregar o projeto e o produto. |
| Jogadores (Usuários Finais) | Público-alvo do produto. Seu interesse reside na jogabilidade, usabilidade e fidelidade às regras do jogo original. |