Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

GGDD: Uma Abordagem Orientada à Engenharia de Requisitos no Domínio de Jogos Digitais

Autores: Gabriela Chaves de Moraes e Lucas Arthur Lermen
Orientador: Prof. Dr. Maurício Serrano

Brasília, DF
2021

Gabriela Chaves de Moraes e Lucas Arthur Lermen

GGDD: Uma Abordagem Orientada à Engenharia de Requisitos no Domínio de Jogos Digitais

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Prof. Dr. Maurício Serrano

Coorientador: Profa. Dra. Milene Serrano

Brasília, DF

2021

Gabriela Chaves de Moraes e Lucas Arthur Lermen

GGDD: Uma Abordagem Orientada à Engenharia de Requisitos no Domínio de Jogos Digitais/ Gabriela Chaves de Moraes e Lucas Arthur Lermen. – Brasília, DF, 2021-

102 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Maurício Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2021.

1. *Game Design Document*. 2. Engenharia de Requisitos. I. Prof. Dr. Maurício Serrano. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. GGDD: Uma Abordagem Orientada à Engenharia de Requisitos no Domínio de Jogos Digitais

CDU 02:141:005.6

Gabriela Chaves de Moraes e Lucas Arthur Lermen

GGDD: Uma Abordagem Orientada à Engenharia de Requisitos no Domínio de Jogos Digitais

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 04 de novembro de 2021.

Prof. Dr. Maurício Serrano
Orientador

Profa. Dra. Milene Serrano
Coorientador

Prof. Dr. Edson Alves da Costa Júnior
Examinador

Brasília, DF
2021

Este trabalho é dedicado à nossas famílias, por sua capacidade de acreditar e investir em nós, e a todos amigos e colegas, pelo incentivo e apoio constantes.

Agradecimentos

Primeiramente, agradecemos a Deus, por conduzir nossa jornada, através de suas bençãos, permitindo-nos chegar a esse momento.

Gratidão também às nossas famílias pelo suporte, apoio e por guiar e acompanhar nossa caminhada. Esperamos poder retribuir toda a confiança depositada em nós.

Agradecemos aos nossos orientadores, Maurício e Milene, pela atenção, auxílio e por nos guiarem durante a elaboração desse trabalho, por meio do conhecimento transmitido.

Por último, mas não menos importante, agradecemos aos nossos amigos, pelos momentos compartilhados durante toda a graduação. Por meio desses, crescemos juntos e nos tornamos pessoas e profissionais melhores.

A todos esses nossos mais sinceros agradecimentos.

"As pessoas mais felizes não tem as melhores coisas.
Elas sabem fazer o melhor das oportunidades
que aparecem em seus caminhos."
(Clarice Lispector, O Sonho)

Resumo

A indústria dos jogos digitais é recente e cresce cada vez mais ao passar dos anos. Acompanhado desse crescimento, surgem as demandas por jogos cada vez mais complexos e prazos cada vez mais justos. Tal fator tem exigido equipes de desenvolvimento cada vez mais multidisciplinares. Visando entregar jogos que gerem divertimento aos jogadores, há um grande foco no aspecto criativo, evitando adotar práticas que, segundo essas equipes, engessam o processo de desenvolvimento do jogo e, por consequência, afetam negativamente a diversão do jogo. Dentre essas práticas que acabam sendo deixadas em segundo plano, estão as englobadas dentro do contexto da Engenharia de Requisitos. Porém, o que é visto como um empecilho, pode ser uma solução, já que a inclusão de algumas práticas consolidadas da Engenharia de Requisitos, durante o processo de desenvolvimento de um jogo, poderia impactar positivamente esse processo, conferindo às equipes de desenvolvimento um melhor entendimento acerca do jogo e seus requisitos. Nesse trabalho, busca-se incluir algumas dessas práticas ao processo de desenvolvimento através do *Game Design Document*, um artefato comumente elaborado durante a concepção de um jogo. Dessa forma, é provido um artefato, chamado GGDD (*Good Game Design Document*), implementando técnicas da Engenharia de Requisitos a ele, sem limitar a expressão dos desenvolvedores e mantendo o fator criativo. Adicionalmente, foi desenvolvido um *website* informativo que orienta os usuários sobre as técnicas implementadas, com exemplos, dentre outros elementos didáticos.

Palavras-chave: Desenvolvimento de Jogos. *Game Design Document*. GDD. Engenharia de Requisitos.

Abstract

The game industry is new and grows more and more over the years. Accompanied by this growth, the demands for increasingly complex games and increasingly short deadlines arise. This factor has demanded much more multidisciplinary development teams. In order to deliver games that are fun for the players, there is a great focus on the creative aspect, avoiding adopting practices that, according to these teams, block the process of game development and, consequently, affect the fun of the game in a negative way. Among these practices that end up being left in the background, are those encompassed within the context of Requirements Engineering. However, what is seen as an obstacle, can be a solution, since the inclusion of some consolidated requirements engineering practices, during the game development process, could positively impact this process, giving the development teams a better understanding about the game and its requirements. In this work, we look for including some of these practices in the development process through the Game Design Document, an artifact commonly elaborated during the design of a game. Thus, an artifact, called GGDD (Good Game Design Document), is provided, implementing Requirements Engineering techniques to it, without limiting the expression of the developers and maintaining the creative factor. Additionally, an informative website was developed. This website guides the users about the implemented techniques, with examples, and others didactic elements.

Key-words: Game Development. Game Design Document. GDD. Requirements Engineering.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

GDD	*Game Design Document*
GGDD	*Good Game Design Document*
IEEE	*Institute of Electrical and Electronic Engineers*
NFR	*Non-functional Requirements*
SIG	*Softgoal Interdependency Graphs*
TCC	Trabalho de Conclusão de Curso
XP	*Extreme Programming*

Sumário

Introdução

Esse capítulo procura auxiliar o leitor em relação a aspectos essenciais do presente trabalho. Buscando realizar tal ação, será apresentada, de forma breve, uma contextualização acerca do tema abordado; sucedida por sua respectiva questão de pesquisa. Ademais, com base no que foi exibido, serão expostos a justificativa, os objetivos e a organização do trabalho.

Contextualização

Jogos digitais não são apenas produtos avançados de software, como também trabalhos complexos de criatividade e arte. Essa fusão de disciplinas torna seu processo de produção um objeto de estudo interessante sob diferentes perspectivas. Concomitantemente, representa diversos desafios para a comunidade de desenvolvimento de jogos (ENGSTRöM et al., 2018).

Tendo em vista que a multidisciplinaridade é uma característica forte na indústria de jogos, grandes projetos contam com times altamente especializados, tendo simultaneamente desenvolvedores de software, *designers*, músicos, roteiristas e vários outros profissionais (PETRILLO et al., 2008).

No intuito de melhorar a compreensão das necessidades e promover a interação entre os times com a menor quantidade possível de falhas de comunicação, faz-se necessário capturar a visão criativa do processo de desenvolvimento do jogo, feita através do *Game Design Document* (GDD). Esse documento é elaborado durante a pré-produção, fase em que há uma espécie de coleta de requisitos por parte dos *game designers* através da elaboração de protótipos e da apresentação da visão criativa do jogo. Durante a etapa de produção, o GDD é utilizado para o desenho, o desenvolvimento e a validação do software. Na pós-produção, os jogos são distribuídos e monitorados após a entrega, com o propósito de realizar ações corretivas e analisar as expectativas da companhia em relação a vendas e desempenho do produto (KANODE; HADDAD, 2009). Logo, desempenha um papel importante durante todo o processo de desenvolvimento de jogos (SALAZAR et al., 2012).

O GDD deve ser minucioso, mas não necessariamente formal, no sentido da estrutura ou de uma perspectiva matemática. Outro aspecto levantado em relação a esse assunto é que impor muita estrutura no processo criativo pode ser altamente prejudicial, restringindo a expressão, reduzindo a criatividade e enfraquecendo os recursos que criam uma experiência divertida para o cliente. De certo modo, é um documento de requisitos de

acordo com a definição do time de pré-produção (CALLELE; NEUFELD; SCHNEIDER, 2005).

No tocante ao GDD, se feito de forma não comprometida e/ou cuidadosa, afeta o escopo do projeto que, consequentemente, impacta negativamente na fase de produção (KANODE; HADDAD, 2009). Erros advindos de uma Engenharia de Requisitos inadequada são custosos e muitos projetos incorrem nessa falha. Além disso, há problemas com o gerenciamento de artefatos, os quais são inerentes ao processo de documentação dos requisitos. Essa realidade não desejada é acordada na literatura (CALLELE; NEUFELD; SCHNEIDER, 2005).

Tendo em vista esses problemas, Kanode e Haddad (2009) defendem que uma aplicação rigorosa dos padrões e processos consolidados da Engenharia de Software pode ser realizada ao longo do desenvolvimento de jogos, com o objetivo de conferir um gerenciamento mais adequado aos projetos, assim como a redução de riscos. Adicionalmente, os autores comentam que essa prática deve ser flexível o suficiente para que as equipes possam utilizá-la mais facilmente.

Dessarte, seria de alta relevância uma proposta de modelo de GDD que valorize o lado criativo, não limitando-o, e que se oriente por princípios da Engenharia de Requisitos, os quais facilitem o gerenciamento do projeto; amparem a transição da pré-produção para a produção, e ainda mitiguem falhas e ameaças.

Questão de Pesquisa

Baseado no que foi contextualizado, e buscando conferir um direcionamento ao desenvolvimento deste Trabalho de Conclusão de Curso, pretende-se responder a seguinte questão de pesquisa:

É possível obter um modelo de GDD, orientado às boas práticas da Engenharia de Requisitos, permitindo aos desenvolvedores de jogos uma compreensão mais adequada quanto às ideias arquitetadas pelos *designers*, durante a fase de pré-produção, a fim de aplicá-las corretamente ao longo das demais etapas do desenvolvimento do jogo?

Justificativa

Em relatórios observacionais com base em práticas da indústria, os autores Callele, Neufeld e Schneider (2005) encontraram poucas evidências de aplicações que de fato se estruturam em princípios da Engenharia de Requisitos, comumente aceitos na revisão por eles realizada.

Conforme Kanode e Haddad (2009), a Engenharia de Requisitos pode auxiliar na

junção dos requisitos claramente especificados e os que não estão tão claramente especificados para o projeto. Isso é feito envolvendo os líderes dos diferentes times (arte, *design*, programação, dentre outros), os quais contribuem para a percepção de que um dado plano é realista ou não.

As melhores práticas de Engenharia de Requisitos podem apoiar os estágios de pré-produção e produção, trazendo estrutura, detalhe, e estabelecendo relações entre os elementos do jogo digital visando ofertar a melhor experiência para o usuário durante o tempo de utilização. Um GDD formal pode fornecer assistência para o processo de transição entre as fases de pré-produção e produção, além de reduzir o retrabalho (SALAZAR et al., 2012).

Segundo Daneva (2014) o GDD é "o pilar de todo projeto de desenvolvimento de jogos". Na pré-produção, esse artefato, ou equivalente, captura o desenho do jogo, comunicando a visão do *designer* para a fase de produção/implementação (CALLELE; NEUFELD; SCHNEIDER, 2011). Adicionalmente, pode auxiliar na especificação e na estruturação dos requisitos (GONZáLEZ-SALAZAR; MITRE-HERNáNDEZ; LARA-ALVAREZ, 2017), servindo como uma base para a Engenharia de Requisitos no processo de desenvolvimento (ALEEM; CAPRETZ; AHMED, 2016a). Por fim, cabe salientar que o GDD pode minimizar problemas de comunicação, os quais são proeminentes no desenvolvimento de jogos (FATIMA; RASOOL; QAMAR, 2018).

A revisão feita pelos autores Callele, Neufeld e Schneider (2011), nas mídias eletrônicas, mostrou que vários autores concordam sobre o que poderia estar em um GDD, mas não necessariamente sobre o que precisa estar nesse documento para apoiar os esforços de produção. Adicionalmente, nenhum autor indicou, efetivamente, como o material deve ser representado no GDD para apoiar os esforços do time de produção. Por exemplo, há um número de *templates* disponíveis para *download*, mas nenhum fornece detalhes substanciais sobre como representar um *design* de jogo digital de maneira que possa ser usado diretamente ou, pelo menos, para gerenciar o esforço de produção (CALLELE; NEUFELD; SCHNEIDER, 2011).

A estrutura básica desse artefato, segundo Aleem, Capretz e Ahmed (2016a), inclui as metas, o gênero e a história por trás do jogo; o fluxo geral, os personagens e seus diálogos; efeitos especiais, número de elementos e *features* que estão presentes nesse produto, e exibe informações, se necessário, de *feature creep*. *Feature creep*, em concordância com Petrillo et al. (2008), é a maior causa de problemas de escopo, sendo ainda uma situação comum, na qual novas funcionalidades são adicionadas durante a fase de desenvolvimento. Consequentemente, essa prática gera aumento significativo no escopo do projeto, dificultando atendimento aos prazos bem como incorrendo em outros efeitos não desejados.

De acordo com os autores Callele, Neufeld e Schneider (2011), se feito de forma

tradicional, aparentemente, o GDD serve mais aos produtores do documento do que aos seus consumidores. Sob a perspectiva da Engenharia de Software, um desafio do desenvolvimento de jogos é elicitar requisitos funcionais a partir do GDD (WANG; NORDMARK, 2015). Tendo isso em vista, ao adicionar boas práticas da Engenharia de Requisitos a um GDD, busca-se facilitar esse processo de elicitação de requisitos, tanto funcionais quanto não funcionais. Sendo assim, almeja-se ir além das funcionalidades, e conferir, adicionalmente, um olhar também centrado nas não funcionalidades.

Objetivos

O objetivo principal do trabalho é propor um modelo de GDD que compreenda princípios da área de Engenharia de Requisitos, evitando reprimir o processo criativo do design de um jogo, e facilitando a transição da pré-produção para a produção. Ademais, comparar com o modelo proposto por Salazar et al. (2012), e desenvolver um *website* explicativo para o modelo. Destrinchando o principal objetivo, chegamos aos seguintes objetivos específicos:

- Definir o modelo de GDD que servirá de base para a proposta;
- Definir as práticas da Engenharia de Requisitos que podem ser úteis no contexto de desenvolvimento de jogos;
- Propor um modelo de GDD;
- Analisar os resultados obtidos, preferencialmente, com especialistas da área de jogos. Deve-se usar, nesse ponto do projeto, pesquisa-ação, com a coleta de dados sendo viabilizada via questionários;
- Comparar o modelo proposto com o sugerido por Salazar et al. (2012), e
- Desenvolver *website* explicativo com base no modelo levantado.

Organização do Trabalho

Este TCC está organizado da seguinte maneira: inicialmente, o capítulo de Referencial Teórico apresenta as áreas e os conceitos relacionados ao tema deste TCC. Em seguida, são expostas, no capítulo de Suporte Tecnológico, as ferramentas e outros suportes utilizados ao longo da realização desse trabalho. Na sequência, são especificadas as escolhas metodológicas, as principais atividades inerentes à realização do trabalho e o cronograma, no capítulo de Metodologia. O capítulo referente ao produto de software (GGDD) deste TCC expõe o modelo final da proposta de GDD e a apresentação das

questões relacionadas ao *site* informativo desse modelo. Em seguida, são apresentados os resultados obtidos na pesquisa, suas análises e as respectivas alterações realizadas a partir desta análise. Por fim, o capítulo de Conclusão realiza um levantamento das principais questões tratadas ao longo do desenvolvimento do TCC, bem como apresenta sugestões para pesquisas futuras acerca do tema abordado.

1 Referencial Teórico

Esse capítulo procura apresentar os principais referenciais teóricos utilizados ao longo do desenvolvimento desse trabalho. O domínio técnico de maior abrangência, no qual o trabalho está inserido, é a Engenharia de *Software*. Sendo assim, o capítulo começa contextualizando esse domínio. Na sequência, e procurando conferir um domínio técnico mais específico para as principais contribuições desse trabalho, tem-se a Engenharia de Requisitos, e atividades, conceitos e técnicas inerentes a essa área.

Mais adiante, há uma seção dedicada as Metodologias Ágeis visando um melhor detalhamento dessas metodologias e, em seguida, é apresentada uma seção dedicada ao domínio de aplicação desse projeto, no caso, o Desenvolvimento de Jogos, sendo as contribuições desse trabalho especificamente focadas em um artefato chamado *Game Design Document* (GDD). Esse documento é comumente utilizado no desenvolvimento de jogos. Melhorias são acordadas para esse documento, orientando-se pelas boas práticas da Engenharia de Requisitos. Por fim, é conferido o resumo do capítulo.

1.1 Engenharia de *Software*

Dada a dependência da sociedade com aspectos tecnológicos, é compreensível apontarmos que o mundo moderno, provavelmente, não existiria sem o *software*. Sistemas computacionais controlam infraestruturas e serviços nacionais, e a maioria dos produtos elétricos incluem um computador e um *software* em sua elaboração. A manufatura e a distribuição industriais são, em sua maior parte, informatizadas, assim como o sistema financeiro, dentre outros domínios de aplicação. *Software* também impacta a área de entretenimento, fazendo-se presente na indústria da música, jogos de computador, cinema e televisão. Portanto, a Engenharia de *Software* é essencial para o funcionamento de sociedades nacionais e internacionais (SOMMERVILLE, 2011).

Em busca de uma melhor compreensão sobre o que é a Engenharia de *Software*, primeiramente, faz-se necessário entender o que é um *software*. Pressman (2011, p. 29) define ambos os termos, conforme segue:

> Um produto que profissionais de *software* desenvolvem e ao qual dão suporte no longo prazo. Abrange programas executáveis em um computador de qualquer porte ou arquitetura, conteúdos (apresentados à medida que os programas são executados), informações descritivas tanto na forma impressa (*hard copy*) como na virtual, abrangendo praticamente qualquer mídia eletrônica. A Engenharia de *Software* abrange um processo, um conjunto de métodos (práticas) e um leque de ferramentas que possibilitam aos profissionais desenvolverem *software* de altíssima qualidade.

Existem vários tipos de sistemas de *software*, desde os simples sistemas embutidos até os sistemas de informações complexos, de alcance mundial. Diferentes tipos de *software* exigem abordagens diferentes e, por esse motivo, não faz sentido procurar notações, métodos ou técnicas universais para a Engenharia de *Software*. Desenvolver um sistema de informações para o meio corporativo e desenvolver um controlador para um instrumento científico são duas abordagens completamente diferentes. Ao mesmo tempo, esses sistemas possuem, por exemplo, grande assimetria com um jogo computacional de gráficos intensos. Nesse contexto, é interessante notar que, apesar dessas aplicações serem bem diferentes entre si, todas precisam de Engenharia de *Software*, embora não necessitem das mesmas técnicas (SOMMERVILLE, 2011).

Sommerville (2011, p. 16) define que "As ideias fundamentais da Engenharia de *Software* são universalmente aplicáveis a todos os tipos de desenvolvimento de sistemas. Esses fundamentos incluem processos de *software*, confiança, proteção, requisitos e reuso". Conforme colocado nessa afirmação, percebe-se que apesar das necessidades serem variadas, dependendo do domínio de aplicação e da intenção do *software* nesse domínio, há princípios, conceitos e práticas da Engenharia de *Software* que auxiliam sempre, tais como: orientar-se por um processo ou metodologia; ter um cuidado maior com os requisitos e a qualidade do *software*, e guiar-se pela reutilização.

Uma técnica utilizada dentro da Engenharia de *Software* para buscar entender melhor os seus produtos é a matriz SWOT. Ela permite estudar competitividade através das seguintes variáveis: Forças, Fraquezas, Oportunidades e Ameaças, conforme apresentando na Figura 1.

Figura 1 – Matriz SWOT

Fonte: (Silva LL, 2009)

As forças são as variáveis internas e controláveis que proporcionam condições favoráveis em relação ao ambiente em questão. As Fraquezas, também internas, inibem a capacidade de desempenho de uma organização ou de um produto, precisando, assim,

serem superadas. Já as oportunidades são situações externas que podem contribuir para a concretização dos objetivos estratégicos. Por fim, as ameaças são situações externas que podem prejudicar a execução de objetivos estratégicos (SOUZA et al., 2013).

Outro desafio da Engenharia de *Software* é conferir aos usuários a melhor experiência de uso possível de seus produtos. Diversas análises foram realizadas, ao longo dos anos, para atingir esse objetivo. Um dos resultados desses estudos foram as heurísticas de usabilidade de Nielsen (1993), que são aplicadas nessa pesquisa.

Diante do exposto, justifica-se o uso dessas práticas, visando mitigar problemas inerentes ao desenvolvimento de *software*, e procurando obter software de maior qualidade, e manutenção facilitada.

No intuito de contribuir para esse desejoso cenário, esse trabalho confere um olhar ainda mais específico, levantando boas práticas da Engenharia de Requisitos, e incorporando-as em um artefato tipicamente consumido por desenvolvedores da área de jogos. Dessa forma, nas próximas seções são conferidos detalhamentos quanto à Engenharia de Requisitos e ao Domínio de Jogos.

1.1.1 Engenharia de Requisitos

Em uma tentativa de aplicar ao desenvolvimento de *software* as melhores e mais relevantes práticas, a Engenharia de Requisitos evoluiu significativamente, ou seja, além de seu escopo original de atuação, englobando grande variedade de técnicas multidisciplinares. Sendo assim, a diversidade de tarefas atualmente realizadas na Engenharia de Requisitos cresceu além do escopo de desenvolvimento de *software* (CALLELE; WNUK; PENZENSTADLER, 2017).

Com essa evolução, Sommerville (2011) define a Engenharia de Requisitos como o processo de compreensão e definição dos serviços requisitados pelo sistema e a identificação de restrições relativas à operação e ao desenvolvimento do sistema. Cabe colocar ainda, conforme esse mesmo autor, que se trata de um estágio particularmente crítico do processo de *software*, uma vez que erros nessa fase, inevitavelmente, geram problemas no projeto e na implementação do sistema. Adicionalmente, a Engenharia de Requisitos procura determinar de maneira precisa as especificações do comportamento do *software*, e a evolução do mesmo ao longo do tempo (ZAVE, 1997).

Zave (1997) define o objeto de estudo da Engenharia de Requisitos como "inerentemente amplo, interdisciplinar e aberto. Aborda traduções das observações informais do mundo real para uma linguagem de especificação".

Essa tradução da observação do mundo real reflete na tarefa fundamental da Engenharia de Requisitos que, de acordo com Leite (1994), é definir os requisitos de um *software*. O próprio autor afirma que, para que esta tarefa seja adequadamente execu-

tada, é necessário o correto entendimento do aspecto de contexto, ou seja, onde realizar essa tarefa, quais os recursos, e quais os limites.

Um requisito é um atributo necessário em um sistema, ou seja, uma declaração que identifica a capacidade, a característica, ou o fator de qualidade em busca de fazer disso algo que tenha valor e utilidade para o cliente ou usuário. Requisitos são fundamentais, solidificando a base para todos os trabalhos de desenvolvimento seguintes. Uma vez definidos os requisitos, os desenvolvedores iniciam os outros trabalhos técnicos, tais como: *design* do sistema, desenvolvimento, teste, implantação e operação (YOUNG, 2004).

Conforme Sommerville (2011), a classificação, frequentemente adotada para os requisitos de *software*, os divide em:

- Funcionais: são declarações de funcionalidades que o sistema deve prover, de reações do sistema mediante entradas específicas e de como o sistema deve se comportar em determinadas situações. Em alguns casos, também podem explicitar o que não deve ser realizado pelo sistema, e
- Não funcionais: são limitações fornecidas pelo sistema em relação aos serviços e funcionalidades por esse oferecidas. Incluem restrições de *timing*, de processo de desenvolvimento e de imposição das normas. Ao contrário das particularidades ou serviços do sistema, os requisitos não funcionais, muitas vezes, aplicam-se ao sistema como um todo.

No intuito de cumprir a tarefa fundamental da Engenharia de Requisito mencionada previamente, é necessário realizar uma série de atividades, definidas por Young (2004), durante o ciclo de vida do sistema.

Primeiramente, é necessário identificar os *stakeholders*, reconhecendo qualquer indivíduo que esteja interessado no sistema ou nas qualidades que precisam ser atribuídas a ele para atender certas necessidades.

Também é preciso obter um entendimento acerca das necessidades dos clientes e usuários para o sistema planejado e das expectativas em relação a esse sistema. Um dos meios de atingir esse objetivo é se utilizar de técnicas como o *Rich Picture*, que consiste de uma representação visual livre da construção de uma ideia inicial de um sistema, sendo algumas de suas utilidades: organizar e fundamentar informações disponíveis, encontrar áreas que necessitam maior aprofundamento, descobrir contradições e refinar as ideias levantadas(MONK; HOWARD, 1998).

Busca-se elicitar e acordar requisitos de diversos tipos. Há várias técnicas de elicitação que podem ser utilizadas nesse momento, conforme apontado em Nuseibeh e Easterbrook (2000). As mais comumente utilizadas são: questionário, entrevista, *storyboarding* e observação, *brainstorming* e introspecção. É importante detalhar as duas últimas para

o entendimento desse trabalho. Segundo Goguen e Linde (1993), a introspecção exige que a pessoa realizando a técnica imagine que tipo de sistema gostaria para determinadas situações. Já o *brainstorming* é uma junção de ideias, frequentemente utilizado para gerar conceitos sobre os problemas e as soluções potenciais para a situação de trabalho (MONK; HOWARD, 1998). A prototipação também é utilizada nesse processo e é, de acordo com Sommerville (2011, p.30) "uma versão inicial de um sistema de software, usado para demonstrar conceitos, experimentar opções de projeto e descobrir mais sobre o problema e suas possíveis soluções".

Visando identificar requisitos, é importante declarar esses em sentenças simples e os separar em conjuntos. Um fator chave no sucesso de um sistema é a abordagem, por parte desse sistema, dos requisitos de negócio, e o auxílio que esse provê a uma organização na finalidade de alcançá-los.

Esclarecer e redeclarar esses requisitos também é necessário, já que confere a conformidade da descrição dos requisitos com a necessidade real do cliente, e verifica se o modo de escrita desses requisitos está compreensível e utilizável para os desenvolvedores.

A tarefa de analisar requisitos busca garantir a correta definição desses requisitos, e realizar verificação e validação a fim de constatar se estão de acordo com o critério estabelecido para um bom requisito.

Outra atribuição essencial é a definição dos requisitos de maneira que possuam o mesmo significado para todos os *stakeholders*. Assim, deve-se investir tempo e esforço significativo nessa tarefa, alcançando o nível de aprender um vocabulário especial ou um léxico, segundo Nuseibeh e Easterbrook (2000), para o projeto, visto que cada grupo de *stakeholders* pode ter uma perspectiva diferente do sistema e de seus requisitos, sendo vital para o sucesso do projeto haver um senso comum entre os interessados. Quando se busca definir os requisitos não funcionais, o NFR (*Non-Functional Requirements*) *Framework* é uma alternativa. Segundo Yrjönen e Merilinna (2009), esse *framework* é baseado principalmente no SIG (*Softgoal Interdependency Graph*), onde cada *softgoal* representa um requisito não funcional do sistema que está sendo desenvolvido. As *softgoals* abstratas são compostas por outras mais detalhadas e concretas. Essa composição pode ocorrer por meio de uma associação *AND* (e), onde ambas as metas são necessárias para o cumprimento da meta pai, ou de uma associação OR (ou), na qual o cumprimento de uma meta já atende as necessidades da meta pai. À medida que as *softgoals* são decompostas, elas se tornam operacionalizações que informam a maneira como cada uma impacta as *softgoals* pai.

Assim como definir os requisitos, é fundamental especificar os requisitos, incluindo todos os detalhes de forma precisa a cada requisito, visando documentá-los e especificá-los em um ou mais artefatos, variando conforme o tamanho do projeto. Um artefato utilizado para realizar esse fim é a *Software Requirements Especification* (SRS). Esse artefato é uma

especificação para um determinado produto de *software*, programa ou conjunto de programas que executa certas funções em um ambiente específico. Ele é uma importante parte no processo de desenvolvimento do *software*, sendo utilizado para *design*, implementação, monitoramento, verificação e validação do projeto e em treinamento. A IEEE recomenda boas práticas para uma melhor especificação dos requisitos de um projeto no documento *IEEE Recommended Practice for Software Requirements Specifications* (COMMITTEE, 1998).

Além disso, faz-se necessário priorizar esses requisitos para economizar tempo e dinheiro realizando tudo o que é desejado no *software*, considerando que os requisitos possuem diferentes níveis de importância, dependendo do interessado em foco. Alguns requisitos são críticos. Portanto, são especificados como sendo de alta prioridade. Outros terão prioridade média, e os demais prioridade baixa. A ideia é lançar o produto com as *features* mais críticas e, futuramente, lançar o produto completo; além de garantir o investimento apropriado para atender as várias necessidades dos interessados. No intuito de orientar o processo de priorização dos requisitos, são utilizadas técnicas específicas. As mais comuns são: MOSCOW (WATERS, 2009) e *First Things First* (WIEGERS, 1999). Apesar de mais antigas, ainda são muito utilizadas no desenvolvimento de software. A primeira técnica é bastante simples, e faz uso de quatro níveis de priorização: *Must* (mais relevante), *Should*, *Could* e *Would* ou *Won´t* (menos relevante). Já a segunda técnica demanda maior conhecimento da equipe, conforme colocado em Wiegers (1999). É possível ir além, e usar técnicas que se apoiam nessas bases, e conferem detalhamentos ainda mais abrangentes, tal como indicado pelos autores Ali et al. (2020); ou ainda orientar-se por uma ferramenta (MOWAD; MUHAMMAD; ELFAKHARANY, 2020).

A partir disso, é possível derivar os requisitos, considerando que o *design* de um sistema pode ocasionar o surgimento de requisitos que atendam às necessidades geradas por esse *design*. Entretanto, esses mesmos requisitos podem não proporcionar benefícios diretos ao usuário final. Por exemplo, um requisito de armazenamento em disco pode resultar da necessidade de armazenar muitos dados.

Tão útil quanto derivar os requisitos é particioná-los, separando em categorias com base na forma como esses requisitos podem ser atendidos. Exemplos de conjuntos: *hardware*, *software*, treinamento, documentação, entre outros. Regularmente, esse processo torna-se mais complexo do que o previsto, devido a alguns requisitos serem atendidos por mais de uma categoria.

Com isso, pode ser feita a alocação desses requisitos, direcionando para diferentes subsistemas e componentes do sistema. Essas alterações nem sempre são satisfeitas por apenas um subsistema ou um componente. Aqui, cabe conhecer técnicas de modularização, desde as tradicionais, baseadas em baixo acoplamento e alta coesão; até mesmo outras propostas mais atuais (ABREU; GOULãO, 2001).

É importante também manter a rastreabilidade dos requisitos verificando o local no qual cada requisito é endereçado. Nesse sentido, torna-se indispensável o rastreamento para as fontes que o embasam (pré-rastreabilidade, ou seja, da *baseline* de requisitos para trás), bem como para outros artefatos que o modelam e/ou implementam (pós-rastreabilidade, ou seja, da *baseline* de requisitos para frente, até o código). Usualmente, essa tarefa é cumprida através do uso de ferramentas automatizadas ou semi-automatizadas que tenham esse propósito, como por exemplo a HelixRM (PERFORCE, 2021).

Para gerenciar esse requisitos elicitados é necessário adicioná-los, apagá-los ou modificá-los durante o ciclo de desenvolvimento do sistema. Adicionalmente ao banco de dados, um conjunto de artefatos, tais como: documentos acerca da visão e do escopo do projeto; lista dos requisitos do projeto; qualquer documento relacionado aos requisitos declarados, entre outros mencionados por Young (2004), compõem o repositório de requisitos. Por fim, cabe colocar que a atividade de gerenciamento de requisitos demanda cuidados com rastreabilidade e versionamento (SAYãO; LEITE, 2006).

Testar e verificar requisitos auxilia na garantia da satisfação por meio da checagem desses requisitos, dos *designs*, dos códigos, dos planos de teste e dos produtos do sistema. Também contribui para garantir que os requisitos estejam em conformidade com as especificações dos mesmos em termos de notações. De acordo com Fagan (2001), inspeções, usando *checklists*, costumam ser realizadas na verificação dos requisitos, com processos bem estabelecidos e realizados pela equipe de desenvolvimento.

Por fim, é preciso validar os requisitos, garantindo que esses requisitos estejam de acordo com as expectativas dos interessados. Testes com a participação dos interessados costumam ser realizados na validação dos requisitos, procurando confirmar se todos os requisitos priorizados para aquela dada entrega estão sendo de fato entregues. Qualquer não conformidade deve ser revista pela equipe, visando o sucesso do *software* na implantação definitiva do mesmo.

Dado que o presente trabalho procura contribuir, aplicando as boas práticas descritas nessa seção no domínio dos jogos, seguem seções mais voltadas a esse domínio de aplicação.

1.2 Metodologias Ágeis

Para coordenar e organizar a execução de projetos de *software* são aplicadas as mais diversas metodologias. Nesta seção, são expostas algumas dessas metodologias, importantes e que impactam no contexto da Engenharia de *Software*.

1.2.1 Scrum

O Scrum é uma metodologia simples, que visa auxiliar pessoas, times e organizações a gerar valor através de soluções com alta adaptabilidade para problemas complexos. Esse método consiste de uma abordagem iterativa e incremental, que busca tornar certos aspectos mais previsíveis e assim ter mais controle sobre os riscos (SCHWABER; SUTHERLAND, 2020).

Um dos papéis-chave do Scrum é o *Scrum Master*, líder responsável por estabelecer o Scrum como definido no guia, e facilitar o entendimento dos envolvidos acerca desse. É tarefa do *Scrum Master* gerenciar o ambiente para o trabalho do *Product Owner*. Esse é encarregado de buscar o maior valor possível do produto resultante do trabalho do time do Scrum. Além disso, transforma atividades complexas em um *product backlog*, uma lista de tarefas necessárias para construir e melhorar o produto final (SCHWABER; SUTHERLAND, 2020).

No início da *sprint*, intervalo de tempo no qual as ideias se tornam um valor, o time do *scrum*, que é composto pelo *scrum master*, *product owner* e desenvolvedores, planeja o *sprint backlog*. Esse último artefato representa as tarefas advindas do *product backlog*, e que serão realizadas durante aquela *sprint*, por meio do evento *sprint planning*. Tem-se, então, um incremento de valor. Durante esse período, ocorre um evento diário, chamado *daily scrum*. *Daily scrum* permite aos envolvidos relatarem o que fizeram; o que pretendem fazer, e os possíveis impedimentos (SCHWABER; SUTHERLAND, 2020).

1.2.2 *Extreme Programming* (XP)

O XP é um estilo de desenvolvimento de *software* que possui foco na aplicação de diversas práticas como técnicas de programação, comunicação clara e trabalho em equipe, além de abordar o risco em todos os níveis do processo de desenvolvimento, gerando, de maneira produtiva, um *software* de alta qualidade (BECK; ANDRES, 2004).

1.2.3 Kanban

É um método simples, de fácil utilização, visualização e adaptação/modificação para gerenciamento de tarefas. Consiste de um quadro com cartões que representam as atividades/tarefas e com colunas definidas conforme a necessidade da equipe e que significam o status daquele cartão (ESPINHA, 2019).

1.3 Desenvolvimento de Jogos

Há meio século, era criado o primeiro jogo em forma de software. Nesse âmbito, diversos aspectos sofreram alterações durante esse período. Recentemente, a indústria

de jogos equiparou-se com outras indústrias bem estabelecidas como música e cinema (ALEEM; CAPRETZ; AHMED, 2016a).

Engström et al. (2018) afirmam que as raízes do desenvolvimento de jogos estão, indiscutivelmente e profundamente, nos campos de desenvolvimento de *software* e de ciências da computação. Jogos digitais têm uma herança em comum com as ciências tradicionais de *software*, tendo em vista que tiveram suas origens nos institutos de Engenharia da Computação, mesmo que tenham seguido um caminho de emancipação gradual.

Desenvolver um jogo nos dias de hoje é uma tarefa de maior complexidade do que realizar o mesmo processo nos primórdios da indústria. Blow (2004) atribuí o tamanho dos projetos e sua complexidade como causas dessa dificuldade. Fato esse que está de acordo com Wang e Nordmark (2015), que apresenta o desenvolvimento no início da era dos jogos digitais como simples, conduzido por times pequenos, onde as arquiteturas de *software* eram feitas de alguns módulos com gráficos 2D, simulação, som, transmissão de entradas e saídas (*I/O*) e o arquivo de código principal (*main*). Já nos dias de hoje, a tecnologia de desenvolvimento de jogos alcança níveis extraordinários.

O mercado de jogos vem em uma crescente. Sua receita aumenta 13% ao ano, sendo que, em 2016, alcançou a marca de $6.7 bilhões. Com base nesses dados, é possível concluir que a tecnologia de jogos está cada vez mais acessível e aceita pelo público de todas as idades. Apesar de ser uma das indústrias que cresce mais rapidamente, possui um processo complexo devido ao seu caráter multidisciplinar (FATIMA; RASOOL; QAMAR, 2018).

Por sua natureza multidisciplinar, o desenvolvimento de jogos apresenta desafios únicos, incomuns, se comparado às aplicações tradicionais de *software*. Os principais desafios encontrados nesse processo são o alto custo e a necessidade de profissionais com vasto conjunto de habilidades (FATIMA; RASOOL; QAMAR, 2018).

Outro ponto que difere uma aplicação tipicamente utilitária e orientada às práticas da Engenharia de *Software* de um jogo, segundo Musil et al. (2010), é o propósito desse jogo, normalmente, buscando ser atrativo e gerar diversão, ocasionando uma diferença significativa nos processos de desenvolvimento. Apesar disso, Lewis e Whitehead (2011) acreditam que tanto a indústria de jogos quanto a Engenharia de *Software* podem trocar valores importantes entre si. Jogos contém uma confluência de propriedades interessantes para o desenvolvimento de *software*, tais como interação em tempo real e uso de componentes de alta capacidade computacional. A Engenharia de *Software*, tendo essas necessidades em vista, auxilia nos problemas que a indústria de jogos possui, procurando apresentar soluções que satisfazem essas rígidas restrições computacionais inerentes aos jogos.

Essa possibilidade de troca de valores deve-se à presença do desenvolvimento de

jogos no processo de evolução do desenvolvimento de *software*, tendo em vista que esses jogos foram influenciados pelas novas práticas de *software*, ao mesmo tempo que contribuíram para elas. Essa indústria é uma parte interessante do desenvolvimento de *software* por diversas razões, e auxilia na criação e na evolução de *software* e *hardware* devido à necessidade computacional. Os jogos estão presentes em diversas plataformas que vão de jogos simples para celular até grandes comunidades de jogos *online* (ENGSTRöM et al., 2018).

Bethke (2002 apud MITRE-HERNáNDEZ et al., 2016) apresentam as três etapas que comumente compõem o processo de desenvolvimento na indústria dos jogos digitais, sendo elas: pré-produção, produção e pós-produção. A pré-produção é o estágio de criação, e se concentra, principalmente, na criação do conceito e no *design* do jogo. Já a etapa de produção, cria e valida o *software*. Nesse estágio, também são produzidas as características exigidas pelo jogo, como sons e música. Finalmente, na fase de pós-produção, são realizadas a distribuição e a manutenção desse jogo, além do gerenciamento de *feedbacks* provenientes de diferentes fontes.

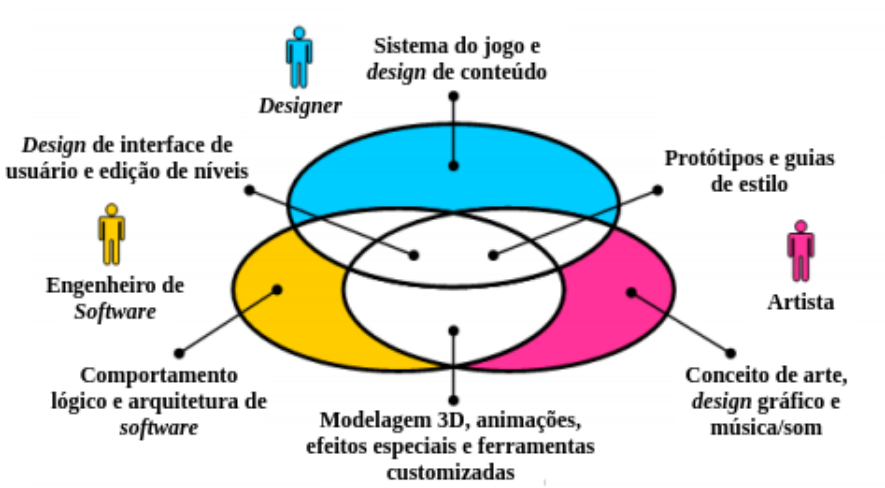
A organização do time é variável, de acordo com a companhia. Frequentemente, a especialidade é o critério de organização para os times, como grupo de programação e um grupo de *design*. Esses grupos podem ter subgrupos como um time de Inteligência Artificial (IA) ou um time para texturas. Cada grupo tem um líder, sendo esse o funcionário mais experiente na área. Esse tipo de organização é um método comum (KANODE; HADDAD, 2009).

Em relação à comunicação entre esses times, Kanode e Haddad (2009) afirmam que combinar esses grupos é positivo para essa questão e aumenta a interdisciplinariedade entre as disciplinas no processo. Unir os diversos grupos pode melhorar o entendimento e a comunicação entre os times, ponto de falha em diversas organizações. A Figura 2 apresenta como funciona essa interação, e demonstra como o desenvolvimento de jogos é baseado em uma mistura de disciplinas com o objetivo de facilitar a comunicação entre as equipes, e o processo de criação de um jogo, gerando um maior benefício para o usuário final.

Em relação aos métodos utilizados pela indústria de jogos, Petrillo e Pimenta (2010) citam que as práticas ágeis parecem incluir as melhores *features* da indústria de jogos, como multidisciplinaridade e dificuldades em aspectos de modelagem, tais como: experiência de usuário, diversão e satisfação; enquanto métodos empíricos são mais adaptativos e reagem melhor às mudanças durante a implementação do projeto.

Métodos e práticas ágeis podem beneficiar o desenvolvimento de jogos de diversas maneiras. De acordo com K.Schwaber e Sutherland (2011 apud KOUTONEN; LEPPäNEN, 2013), um processo iterativo e incremental permite inventar, projetar e testar ideias de um jogo e melhorar sua qualidade. Reconhecimento mais rápido de recursos diverti-

Figura 2 – Desenvolvimento de jogos como atividade colaborativa



Fonte: (MUSIL et al., 2010), traduzidos pelos autores.

dos e implementáveis, e uma melhor comunicação também são benefícios desse tipo de abordagem.

Por esse motivo, times de desenvolvimento de jogos estão adotando práticas ágeis. Esse cenário - a implementação de métodos ágeis como *Scrum* e *XP* - pode ocorrer naturalmente, desde que princípios de agilidade sejam utilizados pelos times em suas rotinas (PETRILLO; PIMENTA, 2010).

Normalmente, esses times unem-se para construir um único jogo, e depois se dispersam. Os treinamentos para essas equipes, em relação a ferramentas de motores de jogos, tem aumentado significativamente. Outro aspecto que chama atenção são os módulos de códigos feitos exclusivamente para alguns jogos. Esses códigos oferecem menos de 30% de reutilização em esforços subsequentes (ZYDA, 2005).

Kanode e Haddad (2009) afirmam que uma fase de pré-produção de sucesso (uma ocorrência rara) é responsável por definir quão emocionante e imersivo é o jogo, ademais, impacta positivamente o estágio de produção, em razão de reduzir a necessidade de encontrar o elemento impreciso “diversão” no decorrer desse período. Possibilita ainda que o time foque na implementação do jogo, ao invés de realizar experiências para encontrar tal elemento.

Ainda na pré-produção, podem ser aplicados métodos ágeis que permitem uma exploração rápida do jogo através do uso de protótipos. Visando reunir todos os requisitos do *design* e determinar um escopo viável, o gerente de projeto deve aplicar práticas da Engenharia de Requisitos e manter a comunicação com os times envolvidos. Deve-se utilizar também a gerência de riscos para tratar do *feature creep* e das mudanças de requisitos

(KANODE; HADDAD, 2009).

Para a fase de produção, é recomendado um processo mais tradicional como o modelo espiral, especialmente, em razão de que a maioria da experimentação acerca da jogabilidade deve ser concluída durante a pré-produção (KANODE; HADDAD, 2009).

As interações entre os requisitos do artefato jogo, os requisitos das ferramentas necessárias para criá-lo, e os fortemente diferenciados grupos de usuários tornam a Engenharia de Requisitos para a produção de jogos digitais particularmente desafiadora (CALLELE; NEUFELD; SCHNEIDER, 2005).

Uma diferença importante entre a Engenharia de Requisitos para jogos e a tradicional é que a elaboração dos requisitos do jogo é muito mais complexa, em razão dos elementos subjetivos, tal como a "diversão", a qual não possui técnicas eficientes para sua determinação. Apresentado esse ponto, Callele, Neufeld e Schneider (2005 apud PETRILLO et al., 2008) afirmam que é necessário estender as técnicas tradicionais da Engenharia de Requisitos, visando apoiar o processo criativo do desenvolvimento de jogos eletrônicos (PETRILLO et al., 2008).

Callele, Neufeld e Schneider (2005) discutem sobre a unicidade da Engenharia de Requisitos para jogos digitais, e apontam que a análise de *postmortems* e de exemplos de jogos revela que essa indústria pode aprender aspectos valiosos das pesquisas e práticas atuais em Engenharia de Requisitos e gestão de projetos. Em seguida, expõe os problemas particularmente notáveis, devido à significância dos mesmos para o sucesso do desenvolvimento de jogos, e sua relevância para a Engenharia de Requisitos, sendo eles: comunicação entre *stakeholders* de diferentes contextos; foco em metas e resistência à *feature creep*; influência de trabalhos anteriores (construir um novo jogo com base em um já existente); interação e integração entre mídia e tecnologia; importância dos requisitos não funcionais, e os requisitos de jogabilidade.

Murphy-Hill, Zimmermann e Nagappan (2014) apresentam suas respectivas visões acerca dos requisitos de jogos como sendo os que o usuário poderá ficar engajado em múltiplas escalas de tempo, e que o meio para atingir tal feito irá variar de jogo para jogo. Esses requisitos, na visão dos autores, também podem partir dos usuários, caso o jogo em questão tenha um predecessor. Entretanto, o requisito "diversão" deve sempre ser o foco principal do jogo e, uma vez que nem sempre o desejo dos usuários gera um aumento dessa diversão, nem todas as sugestões devem ser acatadas e implementadas.

Usabilidade e experiência de usuário são atributos importantes para a qualidade de um *software* "tradicional". Contudo, para o universo dos jogos digitais, a maneira como os usuários interagem com esses produtos demonstra que os requisitos relacionados à experiência de usuário para jogos e os produtos "tradicionais" são diferentes (MURPHY-HILL; ZIMMERMANN; NAGAPPAN, 2014).

1.4 *Game Design Document* (GDD)

Duas mudanças históricas marcaram presença em *design* de jogos, ambas relacionadas a documentos que mediam o desenvolvimento de jogos. Antigamente, os jogos detinham apenas um autor que era responsável por programar as mecânicas do jogo, os gráficos, os sons e a história. Conforme o passar dos anos, os jogos tornaram-se um sucesso comercial, o que gerou um aumento nas expectativas. Além da sofisticação dessas mecânicas e demais elementos. Dessa forma, times maiores foram necessários, e com isso, documentos mais formais e focados no público também foram demandados. O documento de *design* resultante possui uma classificação própria: o *Game Design Document* (GDD) (COLBY; COLBY, 2019).

Aleem, Capretz e Ahmed (2016b) definem o GDD como um entregável crucial da fase de pré-produção. Basicamente, o GDD consiste de uma descrição coerente dos componentes básicos, suas inter-relações, direções e um vocabulário compartilhado para desenvolvimento eficiente de um jogo. Bethke (2002) afirma que esse documento, para a maioria dos *designers*, é uma grande diversão, já que durante a sua criação esses profissionais podem mostrar fortemente sua visão e fornecer a base do conceito do jogo. O propósito desse artefato relembra fortemente o de uma especificação de requisitos da Engenharia de Software tradicional (CALLELE; NEUFELD; SCHNEIDER, 2011).

Em busca de organizar os esforços do time e o processo de desenvolvimento, o GDD é desenvolvido e editado pelo time de *design*. Os diferentes estúdios de desenvolvimento e os variados gêneros de jogos influenciam no formato de um GDD. Frequentemente, é construído aspirando documentar o conceito do jogo e prover uma base para a Engenharia de Requisitos ao longo do processo de desenvolvimento. Em posse desse artefato, os *designers* podem rastrear os esforços de análise de requisitos durante o processo (ALEEM; CAPRETZ; AHMED, 2016a).

O GDD pode articular as partes comuns do processo de *design* e ajudar a prevenir grandes crises, como arte que não é bem renderizada ou *bugs* no final do jogo, além de impactar e ser impactado por *designers* e artistas (COLBY; COLBY, 2019).

Outro papel desse documento é comunicar a visão do *design* para os artistas e programadores e, segundo Colby e Colby (2019), criar uma maneira para que os programadores e artistas comuniquem seu processo de *design* no intuito de dar forma ao GDD. Os próprios autores ainda citam que os programadores e artistas possuem uma relação recíproca com o GDD. Dessa forma, se os artistas não conseguem criar um recurso para uma parte do jogo, eles podem sugerir maneiras de contornar esse problema que leve às alterações no visual do jogo, o que, consequentemente, acarreta em alterações no GDD. Similarmente, se alguma parte do *design* não funcionar quando for codificada ou testada, os programadores podem fazer sugestões para adequar o *design*, impactando no GDD.

Garantir consistência em relação à visão geral do *design* do jogo apresenta-se como o principal propósito do GDD. Possuindo essa relevância, tornou-se uma importante parte do *design* de jogo, tendo em vista que começou a oferecer aos desenvolvedores de jogos um *framework* conceitual para comunicar suas decisões de *design*; uma planta descritiva, com vários níveis de especificidade dependendo da exigência retórica. Ressalta-se que é necessário cautela para documentar cada parte do processo de *design* antes dos esforços de programação começarem (COLBY; COLBY, 2019).

Uma das missões de um GDD é descrever os requisitos funcionais das *features* que estão sendo implementadas no projeto para todos os membros do time. Em sua forma ideal, seria íntegro e passaria por diversas revisões que pretendem deixá-lo mais compreensível e com a jogabilidade ajustada. Em princípio, os desenvolvedores deveriam ser capazes de pegar a sua cópia do GDD e trabalhar tendo ela como base. Na realidade, criar um documento nesse nível retém um alto grau de dificuldade (BETHKE, 2002).

Esse alto grau de dificuldade pode ser comprovado ao analisar o que diversos autores afirmam sobre o conteúdo de um GDD. Segundo Colby e Colby (2019), o conteúdo e o formato de um GDD mudam a fim de atender às necessidades do *design*, e os pontos fortes de cada *designer* influenciam não apenas o conteúdo do GDD, bem como o processo de escrita desse documento. Já Bethke (2002) afirma que tal artefato detalha todos os personagens, os níveis, as mecânicas do jogo, os cenários, os menus, entre outros, ou seja, detalha o jogo. Por fim, Aleem, Capretz e Ahmed (2016a) consideram importante para o GDD um entendimento acerca do desenvolvimento de tecnologias e mídia futuras, de jogabilidade e de outros requisitos não funcionais.

Os GDDs são mais parecidos com planos de negócios do que projetos para um edifício, ou um diagrama de engenharia mecânica, como o *Technical Design Document* (TDD), podendo vir a se tornar uma seção desse GDD, em certos casos (WALFISZ; ZACKARIASSO; WILSON, 2006). Isso ocorre devido ao fato que a indústria não desenvolveu requisitos formais padronizados para um documento de *design* de jogo. Além do mais, é uma indústria jovem, em que a tecnologia envolvida muda rapidamente para estabelecer os requisitos de um GDD. Fatos esses que podem gerar uma série de consequências (BETHKE, 2002).

Como já colocado anteriormente, antigamente, os jogos costumavam ser muito menores em escopo e complexidade, o que tornava mais simples de documentar o *design* do jogo. Portanto, permitia maior informalidade (BETHKE, 2002). E isso é uma das causas de inconsistências nesse documento, que ocorrem, segundo Brown (2010 apud ROUNGAS, 2016), devido aos conceitos serem documentados utilizando termos diferentes e decisões de *design* conflitantes.

Outro problema foi encontrado por Callele, Neufeld e Schneider (2011), que perceberam a presença, no GDD, de descrições que representam o envio de informações

importantes, segundo o time de pré-produção, para o time de produção. Em contrapartida, não notaram evidências de que o GDD dispõe de dados que o time de produção buscaria da pré-produção.

Adicionalmente às colocações acordadas anteriormente, foi sinalizado por Aleem, Capretz e Ahmed (2016b) que, atualmente, o GDD sofre de representação incompleta. Para solucionar esse problema, o desenvolvimento formal desse documento é muito importante, gerando um artefato abrangente, focado no *design* básico do jogo e em suas premissas. O intenção é proporcionar melhorias na qualidade desse jogo.

Por fim, ao finalizar um GDD, é preciso traduzir esse para um plano de projeto. Tal momento, normalmente, é um gerador de problemas. Assim, o gerente de projeto precisa estar preparado para tais desafios (KANODE; HADDAD, 2009). Desse modo, é possível concluir que, apesar da complexidade de escrever um GDD, esse deve ser gerenciado de maneira correta para que os membros do time de produção possam transitar facilmente para a produção do jogo, aumentando consideravelmente a qualidade do produto final que chega ao cliente (ALEEM; CAPRETZ; AHMED, 2016a). Sendo assim, a completude do GDD pode ser medida através de quão bem o seu time foi guiado pelo *design* do jogo (BETHKE, 2002).

1.5 Resumo do capítulo

Neste capítulo, foi apresentada uma visão introdutória da Engenharia de *Software*; seguida por uma explicação sobre a Engenharia de Requisitos, sua evolução, a definição dessa área e maiores detalhes acerca de seu objeto de estudo, além de um conjunto de atividades que auxiliam no cumprimento do objetivo desse trabalho. Esses pontos foram importantes para a criação de um GDD que possui características desses âmbitos.

Em seguida, foi apresentada uma visão sobre o desenvolvimento de jogos, sua história, nuances e características, e um apanhado sobre o GDD, abordando seu conteúdo e as dificuldades de elaborá-lo. Tais tópicos serviram para auxiliar na elaboração do GDD, associado à Engenharia de Requisitos, mantendo a liberdade criativa do documento.

2 Suporte Tecnológico

Esse capítulo procura apresentar as principais tecnologias, entre ferramentas, plataformas e outros suportes, as quais auxiliaram no desenvolvimento desse projeto como um todo.

2.1 Figma

Figma (FIGMA, 2021a) é uma plataforma que auxilia os times a criarem, testarem e entregarem *designs* aprimorados do começo ao fim. Uma das suas formas de uso é a criação de protótipos que conferem uma visão mais concreta sobre o *software* que está sendo produzido, com uso de animações, telas, botões e outros elementos gráficos. Adicionalmente, podem ser definidos fluxos navegáveis, possibilitando uma validação mais criteriosa e adequada junto aos interessados, ou seja, permitindo criar uma planta mais aperfeiçoada do produto de software (FIGMA, 2021b).

Nesse trabalho, o Figma foi utilizado para criação colaborativa, compartilhamento e testagem do protótipo do *site* informativo para proposta de *Game Design Document*.

2.2 HTML 5

HTML (MDN WEB DOCS, 2021b), abreviação para Linguagem de Marcação de HiperTexto em português, é o bloco de construção mais básico do desenvolvimento *web*. É responsável por determinar qual significado e qual estrutura aquele conteúdo possuirá na página. O HTML, normalmente, não é empregado sozinho, sendo utilizado em conjunto com CSS e JavaScript, os quais serão explicados mais adiante no presente documento.

Segundo MDN WEB DOCS (2021c), a mais recente evolução do padrão que define HTML é o HTML5. Essa evolução baseia-se em dois conceitos principais, sendo:

- Nova versão da linguagem HTML, a qual possui novos elementos, atributos e comportamentos, e
- Conjunto maior de tecnologias, o qual viabiliza o desenvolvimento de aplicações e *websites* diversificados.

2.3 CSS 3

CSS (MDN WEB DOCS, 2021a), sigla para *Cascading Style Sheets*, em tradução para português, Folhas de Estilo em Cascata, é uma linguagem de estilo que visa detalhar, por meio da descrição de elementos mostrados na tela, a maneira como um documento escrito em HTML ou em XML (incluindo várias linguagens em XML como SVG, MathML ou XHTML) é apresentado. Atualmente, é uma das principais linguagens da *open web*, sendo padronizada para navegadores *web*, conforme a especificação da W3C.

Ocorreu o emprego dessa linguagem para determinar o estilo de apresentação dos elementos componentes do *site* informativo.

2.4 JavaScript ECMAScript6

JavaScript (MDN WEB DOCS, 2021d), também conhecido como JS, é uma linguagem de programação, a qual tem como características: possuir baixa demanda de poder computacional, ser interpretada, orientada a objetos, e de fácil aprendizagem. Conhecida como linguagem de *scripting* para páginas *web*, está baseada em protótipos, multiparadigma e dinâmica, guiando-se pelos estilos orientado a objetos, imperativo e funcional. Modelagem de comportamento de uma página *web* é uma de suas utilizações.

Nesse caso, foi utilizada para fornecer maior dinamicidade aos elementos que compõem o *site* informativo.

2.5 ZenHub Browser Extension 2.45.79

Mecanismo que se integra diretamente à interface do GitHub, utilizando um único *login*. Sendo assim, o usuário não precisa trocar de interfaces durante o uso, bem como não é necessário realizar um novo cadastro, fatos esses que economizam tempo e auxiliam na manutenção do foco de quem está utilizando. Ademais, é simples de configurar e de integrar. Dentre outros recursos, oferece: quadros que possibilitam transparência, rastreamento e visibilidade das prioridades; dados precisos; certas automações de fluxos de trabalho, e *roadmaps* (ZENHUB, 2021).

Escolhida para auxiliar no registro, no rastreamento e na organização dos *backlogs* do projeto e da *sprint*. Conhecimento prévio, confiabilidade e o atendimento aos requisitos desse trabalho foram critérios que pautaram essa escolha.

2.6 Heroku

É uma plataforma em nuvem que possibilita construir, entregar, monitorar e escalar aplicações. Tem como objetivo prover rapidez no processo de transformação de uma ideia em uma URL, ou seja, um produto, por meio da simplificação dos assuntos envolvendo infraestrutura. (Heroku, 2021)

A simplicidade em relação ao uso, a facilidade no processo de hospedagem do *site*, a integração com o GitHub e a confiabilidade, além da gratuidade, foram aspectos essenciais para a escolha dessa ferramenta como meio de *deploy* para o *site* informativo.

2.7 Resumo do Capítulo

Esse capítulo apresentou as ferramentas e tecnologias utilizadas para apoiar a elaboração do *site* informativo, conciliando os domínios Engenharia de Requisitos e Jogos, mais especificamente, no que tange o artefato GDD. Adicionalmente, também é realizada uma breve explicação sobre cada ferramenta e o motivo de sua escolha.

A Tabela 1 confere uma visão geral dessas ferramentas, suas principais funções e as versões de cada uma delas.

Tabela 1 – Principais ferramentas de apoio

Ferramenta	**Função**	**Versão**
Figma	Prototipação	-
HTML	Estruturação do site	HTML5
CSS	Estilização do site	CSS3
Javascript	Dinamismo de elementos do site	ES6
Heroku	*Deploy* do *site* informativo	-

3 Metodologia

A metodologia é a explicação minuciosa, detalhada, rigorosa e exata de toda ação desenvolvida no método do trabalho de pesquisa. Esse capítulo compreende a classificação da pesquisa realizada nesse trabalho. Conjuntamente, serão apresentados a metodologia utilizada para o desenvolvimento do *site* informativo e outras demandas práticas; a metodologia para análise dos resultados obtidos, e os fluxos de atividades abordando os TCCs 1 e 2. Ao final, tem-se o resumo do capítulo.

3.1 Classificação da Pesquisa

A pesquisa científica é o "resultado de um inquérito ou exame minucioso, realizado com o objetivo de resolver um problema, recorrendo a procedimentos científicos" (GERHARDT; SILVEIRA, 2009).

Gerhardt e Silveira (2009) dividem os elementos de pesquisa em quatro categorias: quanto à abordagem da pesquisa, quanto à natureza, quanto aos objetivos e quanto aos procedimentos. Essa classificação foi utilizada para posicionar a pesquisa deste trabalho.

3.1.1 Abordagem da Pesquisa

A pesquisa em questão tem uma abordagem qualitativa, em concordância com Gerhardt e Silveira (2009), as quais afirmam que essa classificação compreende, de forma concentrada, em um aprofundamento de um grupo social, uma organização, entre outros, ao invés de dados quantitativos. Além disso, as autoras ainda mencionam que uma pesquisa qualitativa faz uso de uma metodologia própria; preocupando-se em explicar o motivo dos aspectos da realidade que não podem ser quantificados, utilizando dados não-métricos e diferentes abordagens para tal feito. Por fim, cabe ressaltar que uma pesquisa de cunho qualitativo busca por resultados, sendo esses os mais fidedignos possíveis.

3.1.2 Natureza da Pesquisa

Esse trabalho é identificado como uma pesquisa aplicada. Portanto, está centralizado em, conforme Gerhardt e Silveira (2009, p. 35), "gerar conhecimentos para aplicação prática, dirigidos à solução de problemas específicos. Envolve verdades e interesses locais". Complementado por Gil (2008), trata-se de uma pesquisa de cunho aplicado imediato, considerando um cenário circunstancial e não no desenvolvimento de teorias de valor universal.

3.1.3 Objetivos da Pesquisa

Em tal âmbito, essa pesquisa está enquadrada em uma pesquisa exploratória, uma vez que atende às definições dessa categoria, conforme proposto por Gil (2008): dispor de maior flexibilidade em relação ao planejamento; ter como meio de realização levantamento bibliográfico e documental, entrevistas não padronizadas e estudos de caso; normalmente, não utilizar técnicas quantitativas em relação à coleta de dados; e possuir como principal objetivo o desenvolvimento, o esclarecimento e a modificação de conceitos e ideias através da formulação de problemas mais precisos ou hipóteses pesquisáveis.

3.1.4 Procedimentos da Pesquisa

No que tange aos procedimentos da pesquisa, foram utilizadas duas classificações: pesquisa bibliográfica e pesquisa-ação.

A primeira foi utilizada para a fase inicial e se trata, segundo Gil (2008), de um estudo realizado por meio de materiais já elaborados e em sua maioria livros e artigos científicos, possibilitando ao pesquisador cobrir amplamente uma série de fenômenos de maneira direta. Fato esse que é de extrema relevância para pesquisas que requisitam dados dispersos pelo espaço. Ademais, nesse tipo de pesquisa, é necessário assegurar as condições de obtenção dos dados e os analisar de forma cuidadosa, além de utilizar fontes diversas.

A segunda foi utilizada no processo de coleta e análise de dados, sendo detalhada na seção de Metodologia de Análise, apresentada ainda nesse capítulo.

3.2 Levantamento Bibliográfico

Com o objetivo de criar uma base de conhecimento e garantir suporte ao presente trabalho, foi realizado um levantamento de diversos artigos que abordam o assunto em estudo. Para isso, buscas iniciais de artigos auxiliaram na identificação de termos essenciais associados ao tema. Ademais, a partir dessas leituras parciais, houve um melhor entendimento do contexto geral do tema.

As principais plataformas selecionadas para a busca de artigos foram a *IEEE*, *ACM* e *Springer*. As *strings* inicialmente utilizadas foram: *"game development"* and *"requirements"* e *"games"* and *"requirements"*. A partir dos resultados obtidos com o uso dessas *strings*, uma análise com base nos principais tópicos presentes em cada texto foi realizada. Ocorreu ainda a centralização, no Zotero, dos artigos que seguiam os seguintes critérios estabelecidos para seleção:

- Estar redigido em língua portuguesa ou inglesa;
- Possuir tema compatível com o assunto em estudo;

- Possuir resumo que compreenda o assunto em estudo, e
- Dispor de tópicos que englobem temática relacionada aos objetivos do estudo.

À medida que o assunto passou a ser mais compreendido, e houve um maior aprofundamento nas características do tema, novas *strings* de busca passaram a ser utilizadas: *"systematic review"* and *"software engineering"* and *"games"*, *"game development process"*, *"software engineering"* and *"computer games"* and *"systematic review"*, *"game development"* and *"software engineering"* e *"game design document"*. Além disso, a análise da bibliografia de artigos já selecionados, bem como a busca por textos de autores conceituados na área, contribuíram para o enriquecimento da base bibliográfica.

Ao final do levantamento, foram contabilizados 77 artigos, sendo 30 da IEEE, 22 da ACM, 11 da Springer e 14 selecionados por outros meios, como a análise da bibliografia de artigos selecionados. Desses 77 artigos, e proferindo uma leitura ainda mais criteriosa nos mesmos, chegou-se a 5 artigos e livros base, sendo: *Proposal of Game Design Document from Software Engineering Requirements Perspective* (SALAZAR et al., 2012), *Requirements Engineering and the Creative Process in the Video Game Industry* (CALLELE; NEUFELD; SCHNEIDER, 2005), *Critical Success Factors to Improve the Game Development Process from a Developer's Perspective* (ALEEM; CAPRETZ; AHMED, 2016a), *Software Engineering Challenges in Game Development* (KANODE; HADDAD, 2009), e *Game Development and Production* (BETHKE, 2002).

3.3 Metodologia de Desenvolvimento

A metodologia de desenvolvimento utilizada foi híbrida, envolvendo a combinação de diferentes metodologias tradicionais, as quais estão detalhadas na seção Metodologias Ágeis presente no Capítulo de Referencial Teórico. Na sequência, tem-se a proposta metodológica deste trabalho apresentada em detalhes.

A metodologia desse trabalho é híbrida, e utilizou conceitos de *product backlog*, *sprint*, *sprint backlog*, *sprint planning*, e *sprint review* advindos do Scrum. Já do XP, a metodologia compreendeu as práticas: programação por pares, refatoração e padronização. Por fim, teve-se o uso do quadro Kanban.

A aplicação deu-se da seguinte maneira: assim que os requisitos estavam consolidados, foi feito o *product backlog*. Esse artefato reuniu todas as tarefas do projeto. A partir desse ponto, iniciava-se o ciclo de *sprints*, as quais tinham duração de duas semanas. No início desse período, era feito o *sprint planning*, visando planejar o *sprint backlog*. Assim, dando início ao desenvolvimento, fez-se uso do princípio do XP, no caso, da programação em pares. Ao final desse intervalo de tempo, era realizada a *sprint review* para avaliar o desempenho daquela *sprint* e mitigar possíveis problemas.

Adicionalmente, foram considerados os conceitos de padronização por meio do uso de folhas de estilo e refatoração do XP. Por fim, ambos os *backlogs*, do produto e da *sprint*, eram organizados no quadro *kanban*, mais especificamente na ferramenta ZenHub Browser Extension 2.45.79. Esse quadro, dentre outras vantagens, permitiu o monitoramento do *status* de cada tarefa realizada no escopo do projeto e pode ser visualizado no seguinte *link*.

3.4 Metodologia de Análise

Visando realizar a análise dos resultados obtidos, foi utilizada a metodologia científica de pesquisa-ação. De acordo com Thiollent (2005 apud GIL, 2008), um tipo de pesquisa com base empírica concebida e realizada com uma ação ou com a resolução de um problema coletivo. Nessa modalidade de pesquisa, os pesquisadores e os participantes representativos da situação ou do problema estão envolvidos de modo cooperativo ou participativo.

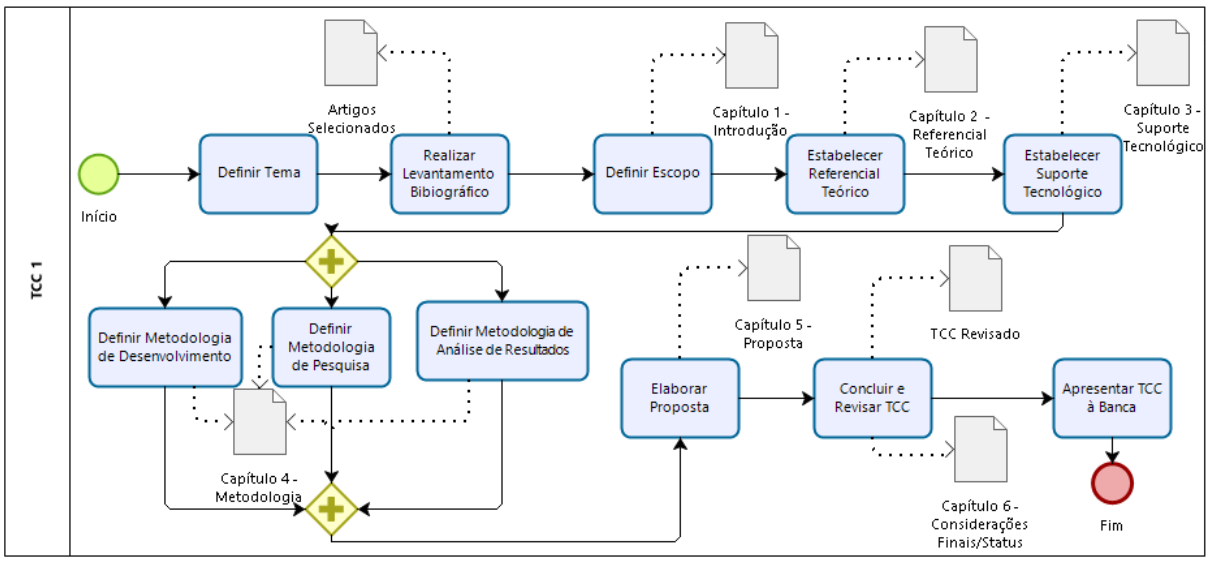
As fases utilizadas estão baseadas em (ENGEL, 2000), sendo elas:

- Definir o problema: compreende a identificação e a definição do problema, avaliando o grau de relevância prática ou mesmo a viabilidade;
- Desenvolver plano de ação: para reverter a situação problemática definida, são consideradas alternativas e opções;
- Implementar plano de ação: quando o plano de ação, esboçado no item anterior, é posto em prática;
- Coletar dados para avaliação: a fim de ter subsídios para realização da análise e da avaliação, os dados serão coletados, e
- Avaliar plano: de posse dos dados levantados na fase anterior, resta analisá-los e interpretá-los, para deles tirar as conclusões cabíveis.

3.5 Fluxo de Atividades

Para a realização de uma pesquisa coerente e dentro do prazo, houve necessidade de organização e planejamento precisos, no intuito de cumpri-la. Visando esse objetivo, este trabalho foi dividido em dois períodos, TCC 1 e TCC 2. Para cada um, foi definido um fluxo de atividades específico, conforme colocado nas subseções a seguir.

Figura 3 – Processo do TCC 1



Fonte: Autores

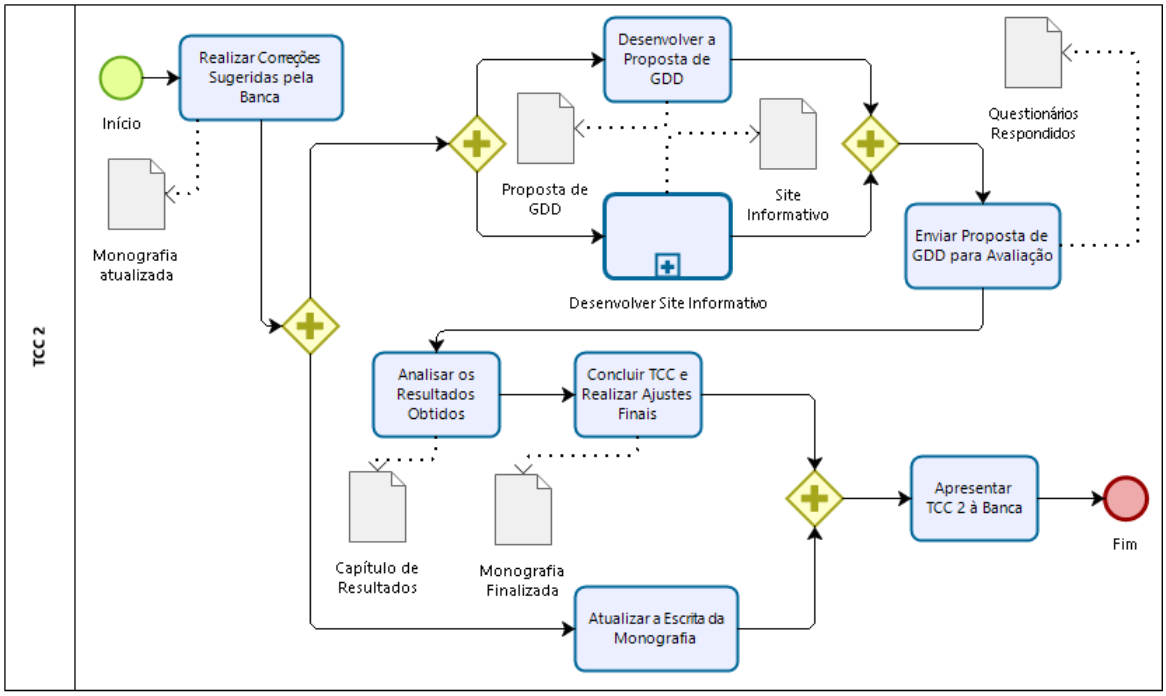
3.5.1 TCC 1

- **Definir Tema:** objetiva escolher o tema que será abordado nessa pesquisa. A escolha foi pautada na existência de *gaps* tecnológicos no levantamento de requisitos na área de jogos;
- **Realizar Levantamento Bibliográfico:** confere uma base de conhecimento acerca do tema escolhido. Foi realizada conforme estabelecido na seção de Levantamento Bibliográfico, e teve como resultado um conjunto de artigos relacionados ao assunto em estudo;
- **Definir Escopo:** com base nos resultados das atividades anteriores, delimita a extensão do que seria tratado nesse trabalho, resultando, dessa forma, na escrita da Introdução;
- **Estabelecer Referencial Teórico:** levanta as principais fontes conceituais que fundamentaram o desenvolvimento do trabalho. Como resultado dessa atividade, tem-se o Capítulo de Referencial Teórico;
- **Estabelecer Suporte Tecnológico:** define e descreve as ferramentas de apoio para o processo de desenvolvimento, culminando no Capítulo de Suporte Tecnológico;
- **Definir Metodologia de Desenvolvimento:** determina os métodos que guiam o desenvolvimento da solução, apresentados nesse capítulo, na seção Metodologia de Desenvolvimento;
- **Definir Metodologia de Pesquisa:** especifica as técnicas metodológicas que norteiam a pesquisa, exibidas nesse capítulo, na seção Classificação da Pesquisa;

- **Definir Metodologia de Análise de Resultados:** detalha o método de pesquisa-ação escolhido para orientar a análise de resultados, como demonstrado nesse capítulo, na seção Metodologia de Análise;
- **Elaborar Proposta:** apresenta um esboço inicial da solução e comprova a viabilidade do projeto, ambos foram detalhados em um trecho do capítulo de Proposta da primeira versão do TCC, disponível no apêndice Modelo Preliminar - Versão TCC1;
- **Concluir e Revisar TCC:** finaliza a escrita da monografia, apresentando as considerações finais. Uma vez de posse desses dados, foi iniciado um processo de revisão, em busca de possíveis erros e melhorias, e
- **Apresentar TCC à Banca:** expõe à banca sobre o que foi desenvolvido durante o período do TCC 1.

3.5.2 TCC 2

Figura 4 – Processo do TCC 2



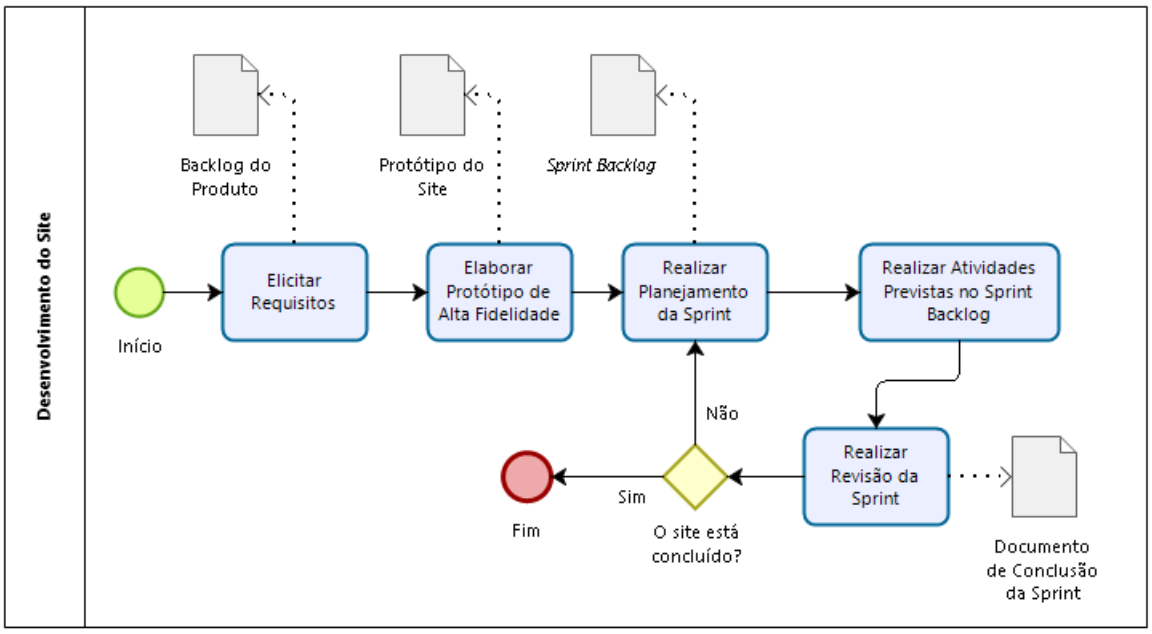
Fonte: Autores

- **Realizar Correções Sugeridas Pela Banca:** realiza correções e evoluções apresentadas pela banca, resultando na monografia atualizada;
- **Atualizar a Escrita da Monografia:** mantém os capítulos previamente escritos em conformidade com as evoluções realizadas durante todo o processo do TCC 2;
- **Desenvolver a Proposta de GDD:** elabora a proposta de GDD que visa solucionar os *gaps* encontrados previamente, resultando no capítulo de GGDD;

- **Desenvolver *Site* Informativo:** constrói *site* no intuito de informar ao público acerca da proposta de GDD;
- **Enviar Proposta de GDD para Avaliação:** encaminha a proposta de GDD bem como questionários para os participantes da pesquisa;
- **Analisar Resultados Obtidos:** examina os resultados obtidos nos questionários, e pondera acerca desses, identificando erros e evoluções. Esse processo é reunido no Capítulo de Análise de Resultados;
- **Concluir TCC e Realizar Ajustes Finais:** completa escrita e revisa esse artefato buscando erros e pontos de melhoria, e
- **Apresentar TCC 2 à Banca:** exibe à banca os resultados finais da pesquisa.

3.5.2.1 Desenvolvimento do *site*

Figura 5 – Processo de desenvolvimento do *site*



Fonte: Autores

- **Elicitar Requisitos:** estabelece os requisitos que irão pautar o desenvolvimento do *site*. Esses requisitos são agrupados no *Product Backlog*;
- **Elaborar Protótipo de Alta Fidelidade:** com base nos requisitos, cria um protótipo de alta fidelidade, representando o *design* seguido ao longo do desenvolvimento do site;
- **Realizar Planejamento da *Sprint*:** planeja, a partir do *Product Backlog*, as atividades realizadas durante a *sprint*, gerando o *Sprint Backlog*;

- **Realizar Atividades Previstas no *Sprint Backlog*:** executa, com base no que foi definido no planejamento da *sprint*, as atividades presentes no *backlog* da *sprint*, e

- **Realizar Revisão da *Sprint*:** pondera acerca de todos os acontecimentos da *sprint*, buscando identificar aspectos que precisavam ser melhorados, pois tiveram impacto negativo; e que podiam ser mantidos pois tiveram impacto positivo. Estão registrados no Documento de Conclusão da *Sprint*, disponível no seguinte *link*.

3.6 Resumo do Capítulo

Nesse capítulo, foram apresentados de forma detalhada os aspectos que constituem a metodologia de pesquisa, desenvolvimento e análise.

Inicialmente, a pesquisa foi classificada de acordo com a abordagem, a natureza, os objetivos e os procedimentos. Em seguida, foram demonstrados os métodos utilizados para o levantamento bibliográfico: as *strings* e bases utilizadas, os meios alternativos para localização dos artigos e os critérios de aceitação desses artigos. Concluído isso, foram especificados os métodos que compõe a metodologia que guiou o desenvolvimento (ou seja, uma combinação de *Scrum*, *XP* e *Kanban*), bem como a maneira que esses foram utilizados em conjunto.

Por fim, foram evidenciados os fluxos de atividades seguidos durante os processos de elaboração dos TCCs 1 e 2, e os cronogramas com as estimativas de prazos para execução dessas atividades.

4 GGDD

Neste capítulo, são apresentados os detalhes do modelo de GDD elaborado e do *site* informativo acerca deste mesmo modelo. Para isso, é exposta uma contextualização com foco na motivação que originou a pesquisa voltada para aplicação de práticas da Engenharia de Requisitos no desenvolvimento de jogos.

Para o modelo de GDD, serão apresentados os modelos base utilizados, bem como métodos, técnicas e artefatos da Engenharia de Requisitos.

Em relação ao *site* informativo, serão apresentadas as telas que compõem este *site*, assim como uma explicação detalhada das opções de *download* do GGDD. Por fim, tem-se o resumo do capítulo.

4.1 Contextualização

A motivação para a escolha do tema surgiu a partir de análises acerca da existência de *gaps* tecnológicos em áreas de interesse comuns aos autores. Uma das primeiras áreas debatidas foi a de desenvolvimento de jogos digitais. Dentro desse tópico, decorreram diversas reflexões no que se referia às falhas neste processo de desenvolvimento. O ponto que mais chamou atenção foi o pouco foco percebido na área de Engenharia de Requisitos e suas práticas.

Tendo isso em vista, foi realizado um aprofundamento maior nos estudos acerca dessa problemática. A partir disso, foram analisados meios de realizar a inserção de práticas voltadas para requisitos no processo de desenvolvimento de um jogo, sem prejudicar a característica fundamental desse processo: a criatividade. Uma das maneiras encontradas foi a incorporação de algumas dessas práticas ao GDD, documento chave, que está presente na maioria dos processos de desenvolvimento e reúne diversos aspectos relacionados à arte, à programação e ao *design*.

Dessa forma, surgiu a proposta do *Good Game Design Document* (GDDD), uma alternativa ao GDD tradicional, e que visa introduzir aspectos da Engenharia de Requisitos no processo de desenvolvimento de um jogo, a fim de mitigar erros. Isso foi realizado buscando não limitar o aspecto criativo e a liberdade dos *designers*.

Juntamente ao GGDD, um *site* informativo, orientando-se por esse modelo, foi elaborado. Tem como objetivo apresentar ao público interessado a ideia, os benefícios e as possibilidades presentes no modelo. Além disso, são disponibilizadas diversas opções de *downloads* para aqueles que quiserem adquirir um entendimento melhor acerca do modelo.

4.2 Base para o modelo

Inicialmente, buscando dar uma sustentação para a proposta, foram pesquisados alguns modelos de GDD já existentes, que serviram para auxiliar na definição do modelo base para o GGDD. As Tabelas 2 e 3 apresentam alguns desses modelos e a sua estrutura. A primeira apresenta um comparativo entre dois modelos propostos: um por Taylor (1999) e o outro por Baldwin (2005). Já o segundo compara dois GDDs utilizados em projetos reais da indústria dos jogos: Silent Hill 2 (COOKE, 2001) e Race'n'Chase (DMA Design Ltd, 1995), um esboço do que viria a ser o primeiro jogo da famosa franquia *Grand Theft Auto*.

É necessário ressaltar também a estrutura e o conteúdo de GDD proposto por Bethke (2002), sendo esse autor uma referência para esse trabalho:

- Seção 1 - Definindo o jogo: contém a articulação sobre o que é o jogo da forma mais compreensível possível e a definição do clima do jogo;
- Seção 2 - *Gameplay* principal: contém qual a principal visão do jogo (ex: 2D, 3D, isométrica); quais as atividades do personagem principal; o diagrama de controle que estabelece as entradas que o jogo irá receber, e a determinação da interface do usuário dentro do jogo;
- Seção 3 - *Gameplay* contextual: contém os diagramas ou outros meios de definição dos menus presentes no jogo e as mecânicas do jogo, dos tutoriais e do *multiplayer*;
- Seção 4 - Comunicar história: contém a história por trás do mundo e dos personagens; o *design* dos níveis, missões e áreas, e as descrições das *cutscenes*, e
- Seção 5 - Cobrir seus recursos: contém as *sprites* 2D ou modelos 3D, listas com as missões, os níveis ou as arenas com alguns parâmetros como tamanho, prioridade, hierarquia, entre outros; a descrição acerca das vozes necessárias para o jogo; a captura de movimentos, e o *key framing*, que são as descrições relacionadas às animações. Ademais, estabelece os efeitos sonoros, a música e os efeitos especiais.

4.3 Modelo final do GGDD

Usando como referência os modelos citados anteriormente, chegou-se ao modelo inicial do GGDD, apresentado no Modelo Preliminar - Versão TCC1. Em cima desse modelo preliminar, diversas análises foram realizadas e mudanças foram feitas nesse modelo. Todas essas análises e mudanças estão especificadas no capítulo de Análise de Resultados.

Por fim, chegou-se à proposta final do GGDD. A seguir serão apresentadas as seções sugeridas para esse modelo. É importante destacar que cada jogo possui um con-

Tabela 2 – Principais ferramentas de apoio - Primeiro Comparativo

Exemplo GDD Taylor (1999)	**Exemplo GDD Baldwin (2005)**
1. Nome do Jogo	1. Página de título
2. Histórico de Versão	2. Histórico de Versão de *Design*
3. Visão Geral do Jogo (Filosofia e questões comuns)	3. Visão Geral do Jogo (Conceito, escopo, gênero, público-alvo...)
4. Conjunto de características (Gerais, *Gameplay*...)	4. *Gameplay* e Mecânicas (Objetivos, estrutura de *puzzles*, físicas, movimentos...)
5. Mundo do jogo (Visão geral, câmera, *Engine*...)	5. História, Configuração e Personagem (Narrativa, *CutScenes*, personagens, mundo do jogo...)
6. *Layout* do mundo (Visão geral e detalhes)	6. Níveis (Definir os níveis e alguns de seus aspectos como sinopse, objetivos...)
7. Características dos personagens (Criação, inimigos...)	7. Interface (Áudio, música, câmera, menus...)
8. Interface de usuário (Visão geral e detalhes)	8. Inteligência Artificial (Tipo, configurações...)
9. Armas(Visão geral e detalhes)	9. Tecnologia (*Hardware* alvo, desenvolvimento de *software*, *engine* do jogo...)
10. Música e efeitos sonoros (Visão geral, *design* de som...)	10. Arte do jogo (arte do conceito, dos personagens, dos ambientes, entre outros, guias de estilo...)
11. Jogo *Single Player* (Detalhes, história...)	11. *Software* Secundário (Editor, instalador, atualizações...)
12. Jogo *MultiPlayer*(Customização, servidores...)	12. Gerência (Calendário, plano de teste, orçamento, análise de risco...)
13. Renderização de personagem (Visão geral e detalhes)	Apêndices
14. Edição de mundo (Visão geral e detalhes)	-
14. Miscelânea (Visão geral e "Lixo" em produção)	-
Apêndices	-

texto próprio e, portanto, o modelo poderá ser adaptado para melhor adequação em cada contexto. O modelo foi exemplificado no apêndice Versão Instanciada - Overcooked 2.

4.3.1 Capa

A capa do GGDD é composta pelas seguintes informações: Nome do Autor/Empresa, Nome do Jogo, Versão, Informações de *Copyright* e Data. Esses elementos são sugeridos pelos autores, porém o usuário pode realizar alterações para atender suas necessidades.

Tabela 3 – Principais ferramentas de apoio - Segundo Comparativo

Exemplo GDD de Race'n'Chase (DMA Design Ltd, 1995)	**Exemplo GDD de Silent Hill 2 (COOKE, 2001)**
1. Capa	1. Capa
2. Histórico de Revisão	2. Sumário
3. Sumário	3. Conceito do jogo(Introdução, descrição...)
4. Introdução (Breve introdução, escopo do documento e convenções de tipografia)	4. Mecânicas de jogo (fluxo, personagens, monstros, elementos de *gameplay*...)
5. Referências	5. Interface (Requisitos funcionais, telas *"mockadas"*)
6. Sistema Alvo (Plataformas alvo e suas descrições incluindo versões, características de *hardware*, etc)	6. Arte e vídeo (Arte, animação, cinemática...)
7. Sistema de Desenvolvimento (Características e configurações do ambiente de desenvolvimento)	7. Som e música (Objetivos gerais, efeitos sonoros e música...)
8. Especificação (Conceito, estrutura do jogo, jogadores, história, objetivos...)	8. Visão geral dos níveis (Localização e desafios)
9. *Gameplay* (Características do mundo, do ambiente, da superfície, dos objetos, de controle...)	9. Análise de mercado (Mercado alvo, comparação de características...)
10. *Front End* (Abertura e menus do jogo)	10. Referências bibliográficas
11. Ferramentas de Desenvolvimento (Características do editor dos *arrays* 3D)	-
12. Time (Função e o nome da pessoa que a desempenha)	-
13. Tempo (Datas de início e fim do projeto, da conclusão do *Game Design* e das *milestones*)	-

4.3.2 Sumário

Esta seção apresenta os títulos de todas as seções e subseções do documento, bem como as páginas que se iniciam, para auxiliar, caso se esteja buscando algo específico no GGDD.

4.3.3 Histórico de Versão

Esta seção deve ser atualizada cada vez que alterações forem realizadas no documento, gerando uma nova versão dele. Nesta seção, há um resumo do foco das mudanças,

bem como uma lista das alterações realizadas.

A não utilização dessa seção não impacta negativamente no desenvolvimento desse documento, mas seu uso é uma sugestão para uma maior rastreabilidade acerca das mudanças do documento.

4.3.4 *Brainstorming* e Introspecção

Registrar as ideias levantadas acerca do jogo é uma excelente maneira de manter o rastro do que já foi imaginado para esse jogo. Para isso, foi introduzida essa seção, dividida em duas partes.

A primeira delas é referente ao registro de ideias abordadas ao longo da concepção da ideia do jogo. Basicamente, sempre que alguma ideia for proposta, seja ela incluída na proposta do jogo ou não, essa ideia deve ser apresentada nesse tópico.

Já a segunda parte baseia-se na técnica de introspecção, na qual as pessoas responsáveis pela concepção do jogo devem se colocar no papel dos jogadores e imaginar quais aspectos esses jogadores desejariam que o jogo possuísse. Tudo isso deve ser registrado nesse tópico.

A ideia dessa seção é estar disponível para a consulta sempre que necessário, permitindo que sejam verificadas todas as ideias já sugeridas para o jogo.

4.3.5 Visão Geral do Jogo

Seção que apresenta um panorama geral das principais características do jogo. Inicialmente, há um tópico para realizar uma breve descrição do jogo, seu tema e foco principal, e em seguida, deve ser apresentado de maneira rápida o mundo do jogo e suas características principais, ou seja, o que há de mais marcante no jogo.

Seguindo pela seção, são encontrados os tópicos, no quais o gênero do jogo deve ser explicitado, sucedidos pela subseção para apresentação do público alvo. Deve ser mostrado o escopo do projeto para que se tenha uma noção do tamanho que o jogo irá possuir. Nesse escopo, são apresentadas as quantidades de níveis, localidades, inimigos, personagens não-jogáveis (NPCs), entre outros fatores que possam influenciar na magnitude do jogo.

Por fim, a utilização de um *Rich Picture* proporciona uma melhor visualização dos principais pontos do jogo, e pode contribuir para uma melhor compreensão da proposta em sua totalidade.

4.3.6 Fatores Competitivos

O objetivo dessa seção é apresentar os motivos que farão do jogo um sucesso. Inicialmente, deve ser apresentada uma matriz SWOT para que se possa analisar as

forças e fraquezas do projeto, além de compreender possíveis oportunidades e ameaças que podem afetar positivamente ou negativamente o desenvolvimento do jogo.

Há ainda um tópico para fazer comparação em relação aos concorrentes. Nesse tópico, a sugestão é que se escreva os principais jogos de proposta semelhante, e explique por qual motivo o jogo em desenvolvimento será superior. Caso seja de interesse dos responsáveis pelo desenvolvimento do jogo, é possível realizar um *benchmark*. Um exemplo de *benchmark* é apresentado no artigo de Oliveira, Miranda e Gomide (2018). Entretanto, isso aumentaria bastante a complexidade do documento.

4.3.7 *Gameplay* e Mecânicas Principais

Esta seção é uma das mais importantes do documento, na qual são apresentadas os principais aspectos para o funcionamento do jogo. Na parte relacionada à *gameplay*, devem ser apresentados os objetivos que os jogadores devem alcançar durante o jogo, bem como a maneira que a progressão ocorre. É importante ressaltar que, caso o jogo possua mais de um modo, essas características devem ser explicitadas para cada modo.

Em relação à mecânica do jogo, deve ser feita, inicialmente, uma descrição de como funciona a física do jogo, seja ela muito simples ou extremamente complexa. Nesse tópico, é interessante explicar como os elementos interagem entre si, ou individualmente, em relação a esse aspecto, bem como apresentar a forma como o universo do jogo trabalha essas questões. Em seguida, encontram-se os tópicos de ações dos personagens e ações do jogo. O primeiro deve ser composto de uma descrição de todas as ações que podem ser realizadas pelos personagens principais do jogo, tais como: andar, correr, mirar, atirar, entre outras. Já o segundo, deve descrever todas as ações que não são realizadas especificamente por personagens, mas afetam a *gameplay* do jogo. Por exemplo, em um jogo onde o jogador controla um exército, ações de movimentar as unidades e selecionar alvos não são exclusivas de personagem algum, porém são ações base para esse tipo de jogo.

Há um tópico para o combate do jogo, caso esse esteja presente. Nesse tópico, são apresentados os inimigos do jogo, com suas principais características (pontos de vida, estilo de combate, dano infligido, esboço da arte, dentre outros). Além disso, também serve para explicar o sistema de combate, apresentando as maneiras de causar ou sofrer danos, a forma de bloquear e desviar de ataques, entre outros diversos fatores que podem afetar esse sistema de combate.

Consta um tópico para demonstrar o sistema de economia do jogo, seja ele simples ou complexo, além de demonstrar quais fatores influenciam nessa economia.

Por fim, existe um tópico para apresentar todas as entradas e os comandos que o jogo pode receber, inclusive para abrir menus, sair de partidas, entre outros. Para esse tópico, o uso de algo mais ilustrativo é recomendado, facilitando a comunicação e a

visualização.

4.3.8 Aspectos de qualidade

Seção inspirada no modelo conceitual do ***NFR Framework***, consistindo de uma adaptação, incluindo o nome, para oferecer maior familiaridade ao usuário. Utilizada em formato de tópicos e texto. Primeiramente, é necessário mentalizar quais aspectos de qualidade (requisitos não funcionais) deseja-se alcançar com o jogo, por exemplo:diversão, desafio, entre outros. Esses serão considerados os macro critérios. Deve-se refletir se algum outro subtópico de qualidade é necessário para cumprir com o macro critério estabelecido, mesmo que de forma satisfatória e não total. Esses seriam considerados critérios de níveis intermediários, sendo opcionais. Por fim, é preciso ponderar sobre maneiras/ações para atingir o aspecto de qualidade estabelecido. Esse último nível compreende as operacionalizações, sendo essas formas concretas para viabilizar o cumprimento dos critérios de níveis superiores.

4.3.9 História

Essa seção é divida em três macro tópicos, sendo eles: Narrativa, Personagens e *Cutscenes*. A Narrativa está dividida entre *Background* da História, que visa explicar o contexto da história, o foco da narrativa e os principais pontos de virada da história do jogo, e Encerramento da História que busca descrever como a história se encerra, e, no caso da existência de múltiplos finais, detalhar cada um.

Em Personagens, tem-se a captura das informações de cada personagem, assim, o subtópico é denominado pelo nome do personagem e apresenta a história, a personalidade, as características(físicas, com possibilidade de inclusão de esboço ou arte do personagens e/ou que influenciam na *gameplay* como pontos de vida, dano causado, entre outros), o papel na narrativa e sua relação com os outros personagens.

Em *Cutscenes*, propõe-se o detalhamento das *cutscenes* presentes no jogo, logo, o subtópico, assim como na subseção anterior, é denominado pelo nome da *cutscene*, e demonstra os personagens, a descrição e o *script*.

4.3.10 Níveis e/ou mundo do jogo

Esse ponto é divido em duas grandes áreas: Níveis e Mundo do Jogo. Em Níveis, cada subtópico leva o nome dos níveis, sendo realizado um resumo que seria uma breve descrição do ambiente e das tarefas que serão desempenhadas; são apresentados os objetivos que precisam ser cumpridos nesse nível; são ilustrados e descritos o mapa, o caminho principal do jogado, os caminhos alternativos (caso o jogo possua essa abordagem) e a conclusão com os acontecimentos após a finalização da fase.

Tratando do Mundo do Jogo, esse apresenta a visão geral do mundo, que demonstra e descreve o mundo do jogo e suas principais características. Após isso, são definidas as áreas do jogo, detalhadas por meio de uma descrição geral do ambiente e das tarefas desempenhadas naquela área, de um mapa, das características específicas que determinam as particularidades daquele ambiente, das conexões com outras áreas e como são feitas, e dos níveis/missões que utilizam daquela região.

Para esta seção, é recomendada a técnica da prototipação para definição dos mapas e da interação entre as diversas áreas do jogo.

4.3.11 Interface

Esse tema consiste de três subtópicos, sendo eles: Menu, no qual são apresentados e detalhados os menus presentes no jogo, como menu principal, menu de pausa, entre outros; *Head-Up Displays (HUDs)*, que demonstram e especificam os HUDs presentes na tela, durante a *gameplay* e, por fim, os Sistemas de Ajuda (Tutoriais), os quais pormenorizam a forma como os tutoriais serão exibidos para os jogadores.

Recomenda-se a utilização da técnica de prototipação para fornecer melhor visualização desses aspectos.

4.3.12 Arte e vídeo

Essa seção é segmentada em Conceito de Arte, que objetiva explanar o objetivo da proposta de arte do jogo, apresentando as bases dessa arte e as reações que se busca provocar nos jogadores através dela, além de apresentar as principais características da arte do jogo. Por fim, há tópicos para listar os elementos do jogo que são elementos 2D, 3D ou cinemáticas.

4.3.13 Música e efeitos sonoros

A seção está dividida em três áreas: proposta do *design* de som, músicas e efeitos sonoros. Em *design* de som, busca-se explicar o objetivo desse *design* no jogo, apresentando as bases e quais as reações que se busca provocar nos jogadores através dele. Além disso, apresentar as principais características desse *design*.

Em Músicas, são explicados o impacto que a música do jogo busca causar aos jogadores, as reações esperadas, e como essas músicas se relacionam com a temática. Pode-se incluir a lista de músicas que compõe a *soundtrack* do jogo.

Em Efeitos Sonoros, são citados os principais efeitos sonoros e, se houver, os objetivos que se busca atingir com eles. Outra alternativa seria simplesmente explicitar como os efeitos sonoros serão trabalhados em conjunto com a temática do jogo.

4.3.14 Sistemas Alvo

Seção dedicada à apresentação das plataformas que irão executar o jogo, bem como dos requisitos de *hardware* mínimos necessário para tal.

4.3.15 Cronograma

Esse tópico inclui a data de início do projeto; as *milestones*, que devem possuir uma breve descrição do marco e o que o compõe; uma breve explicação dos objetivos e o prazo de finalização; a data de conclusão do projeto e, por último, a priorização que ocorre através da utilização da técnica MoSCoW. Essa última registrada em forma de tabela, que consiste em levantar todos os requisitos necessários, listá-los e eleger uma prioridade entre as disponíveis: *Must* (nível mais alto de prioridade, requisitos críticos), *Should* (prioridade média, requisitos essenciais, porém não críticos), *Could* (prioridade baixa, requisitos importantes, mas que não precisam ser atendidos com certa urgência), *Would* (prioridade mais baixa, seriam funcionalidades que complementariam o produto, porém não são vitais).

4.3.16 Referências

Atualmente, grande parte dos jogos utiliza diversos aspectos baseados ou semelhantes a outros jogos. Fato esse que justifica a existência dessa seção. Ela contribui para que tudo o que for baseado em diferentes jogos seja referenciado. Assim sendo, sempre que alguma característica tiver como base um jogo já existente, deve-se colocar a referência para esse jogo no texto e explicitá-lo nessa seção.

4.4 *Site* Informativo

Juntamente ao GGDD, o *site* informativo sobre esse modelo foi elaborado. O objetivo buscado foi apresentar ao público, interessado na proposta, os benefícios e as possibilidades presentes. Além disso, disponibilizar um modelo em branco (ou seja, um *template* ou modelo base), sendo esse um modelo de exemplo, instanciado a partir do *template* e preenchido pelos autores, para auxiliar no entendimento em relação às particularidades do GGDD e uma versão parcialmente preenchida através de um formulário. Por fim, serão apresentadas as telas e breves explicações sobre essas. O *site* está disponível para acesso em *link*, e o código fonte está armazenado no seguinte repositório.

Em termos de *design*, foi percebido que alguns elementos estabelecidos no protótipo realizado não estavam adequados ao que se desejava para o *site* e esses foram evoluídos. Tratando-se da paleta de cores (Figura 6), essa permanece igual. Em relação à tipografia, a EB Garamond foi alterada para a Arial, devido à uma complicação no

uso da fonte, percebida durante a construção do *site*. Essa complicação se refere à falta de compatibilidade dessa fonte com alguns *browsers*. No que tange à usabilidade, foram aplicadas as heurísticas que foram estabelecidas previamente, sendo elas:

- *Design* simples e natural;
- Falar a linguagem do usuário;
- Minimizar carga na memória do usuário;
- Consistência e padrões, e
- Saídas.

Figura 6 – Paleta de Cores

Fonte: Autores

A tela principal, ilustrada na Figura 7, é responsável por receber os usuários. É apresentada uma imagem relacionada ao contexto que a proposta está inserida; o título e uma breve introdução do modelo, e um botão que permite o usuário iniciar sua experiência, redirecionando-o para a tela de *Download*. Logo abaixo, são apresentados os diferenciais contidos no GGDD e uma rápida descrição desses. Além disso, está disponível uma barra de navegação que permite ao usuário acessar as telas de Proposta, *Download* e Sobre.

A tela de Proposta (Figuras 8 e 9) apresenta um resumo de todo o documento, passando por breves explicações sobre cada uma delas, além de alguns exemplos para que o usuário fique familiarizado com o modelo.

A tela de *Download* (Figura 10) apresenta ao usuário as três opções que ele possui para baixar o GGDD. Na primeira delas, está a estrutura da proposta, com uma explicação sobre cada seção do documento. Já na segunda, o usuário será redirecionado para uma tela de formulário, onde ele deve preencher algumas informações iniciais sobre o seu jogo. Assim que tudo for preenchido e a opção de *download* for selecionada, será baixado um arquivo de texto com os dados preenchidos juntamente com o documento *Word* da estrutura, para que o usuário possa inserir esses dados na estrutura do seu GGDD, ou até mesmo compartilhá-los com o resto da equipe. Por fim, há uma opção para o usuário fazer o *download* de uma versão instanciada, com o jogo *Overcooked 2*, disponível em Versão Instanciada - Overcooked 2.

Figura 7 – Tela Principal

Fonte: Autores

Figura 8 – Tela Proposta - Parte 1

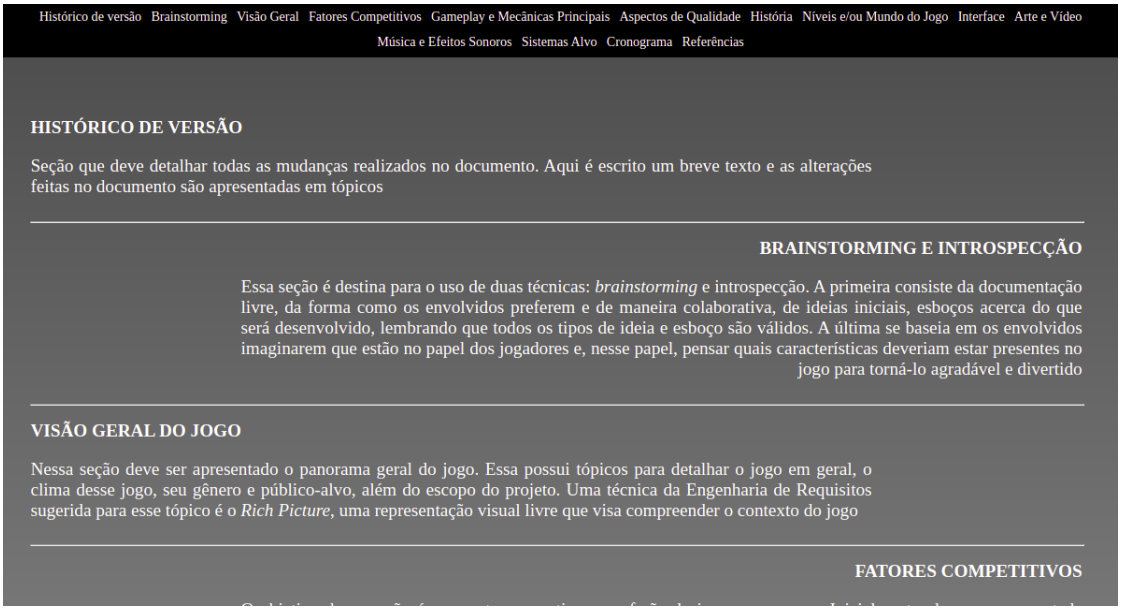
Fonte: Autores

A tela de Sobre, representada na Figura 11, objetiva fornecer informações adicionais sobre o GGDD para o usuário que esteja interessado em maiores informações sobre o modelo, sendo essas: definição e objetivos do GGDD, bem como dados sobre os desenvolvedores.

4.5 Resumo do Capítulo

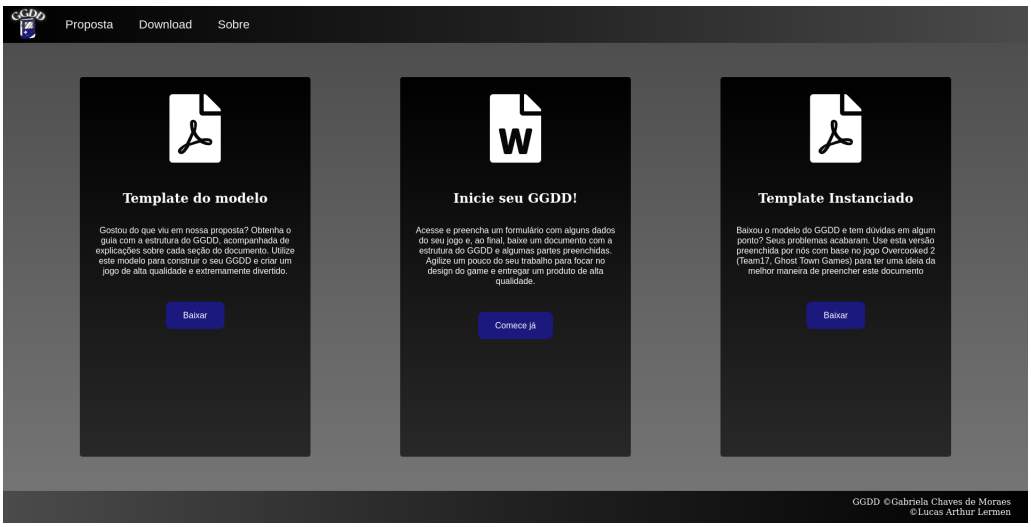
Nesse capítulo, foi explicado sobre o produto de *software* obtido com a realização dessa trabalho. Inicialmente, o contexto dessa proposta foi apresentado, explicitando a motivação e as razões para a escolha do tema.

Figura 9 – Tela Proposta - Parte 2



Fonte: Autores

Figura 10 – Tela *Download*



Fonte: Autores

Em seguida, foram detalhadas as duas partes que compõem esse produto de *software*. A primeira parte refere-se à proposta do GGDD, na qual foi detalhado o modelo base utilizado, e apresentada a versão final da solução, com uma explicação de cada seção dessa versão. A segunda parte aborda questões relacionadas ao *site* informativo, apresentando suas telas, o funcionamento, e os objetivos de cada uma delas.

Figura 11 – Tela Sobre

Fonte: Autores

5 Análise de Resultados

Esse capítulo apresenta a análise dos resultados obtidos ao longo dos ciclos de pesquisa-ação, realizados durante esse Trabalho de Conclusão de Curso, bem como as ações tomadas com base nessa análise, visando melhorar o modelo proposto para o GGDD. Por fim, tem-se o resumo do capítulo.

5.1 Atividades de pesquisa-ação

Como estabelecido em Metodologia de Análise, foram previstas as seguintes atividades: definição do problema, desenvolvimento do plano de ação, implementação do plano de ação, coleta de dados para avaliação e avaliação do plano.

A primeira dessas atividades (definição do problema) foi realizada com base no estudo do material que fundamenta a pesquisa deste TCC para entendimento do *gap* tecnológico acerca de requisitos no domínio de jogos digitais.

As demais atividades estão presentes dentro dos diversos ciclos, e se repetiram a cada novo ciclo iniciado.

5.2 Primeiro ciclo - Autoavaliação

Este ciclo tem como objetivo analisar o modelo preliminar, disponível no Apêndice Modelo Preliminar - Versão TCC1, e identificar possíveis pontos de melhoria. Uma vez que se trata de uma crítica à algo autoral, a etapa de coleta de dados, inerente à metodologia de análise prevista, não foi realizada.

O plano de ação desenvolvido nesse ciclo compreende uma revisão, realizada pelos próprios autores, do modelo inicialmente proposto e, segundo os conhecimentos obtidos durante a elaboração do referencial teórico, a correção de pontos que poderiam causar empecilhos na utilização do GGDD.

Este plano de ação foi implementado e, de acordo com análise feita pelos autores, foram identificados alguns problemas e suas respectivas alterações visando à melhoria do modelo

O primeiro aspecto notado trata-se do tópico **Visão do Jogo**, pertencente à seção de ***Gameplay* e mecânicas principais**. Esse tópico foi retirado, pois não agregava informações muito úteis à seção que se encontrava presente. Os demais itens dessa seção passam por todos os seus aspectos importantes, e apresentam uma boa ideia acerca das principais mecânicas e dos elementos de *gameplay* do jogo.

Inicialmente, foi pensado em um tópico para explicitar as ações realizadas pelo personagem principal, na seção de ***Gameplay* e mecânicas principais**. Porém, ao fazer uma análise dos diversos gêneros de jogos presentes na indústria, foram encontrados alguns pontos de melhoria. Por exemplo, existem jogos no modelo de construção de cidades, que não possuem especificamente personagens que realizam ações. Assim como muitos jogos de estratégia possuem esses personagens, mas contam também com uma vasta quantidade de ações que não são praticadas por eles. Dessa maneira, visando abranger a maior quantidade possível destes gêneros, o tópico **Ações do Personagem Principal** foi substituído por outros dois: **Ações dos Personagens** e **Ações do Jogo**.

Em diversos jogos, ao mesmo tempo que certa unidade (por exemplo, um grupo de soldados de um exército) pode ser inimiga durante um combate, ela pode ser controlada pelo próprio jogador no combate seguinte ou até no mesmo combate. Dessa forma, buscou-se capturar essa possibilidade, expandindo o conceito do tópico de **Inimigos**, na seção de ***Gameplay* e mecânicas principais**, para unidades em geral, caso o jogo tenha essa proposta. Porém, caso o jogo em questão possua apenas inimigos, o tópico é mantido para esse contexto.

Na versão preliminar, estava prevista uma seção nomeada **NFR** que é a sigla para *Non Functional Requirements*. Entretanto, na revisão, foi percebido que esses termos possuíam maior popularidade na área de *Software*, e que alguns usuários da proposta poderiam não estar familiarizados com os termos, assim dificultando a compreensão de tal seção. Logo, sua nomenclatura foi trocada para **Aspectos de Qualidade**, visando facilitar o entendimento e transmitir o objetivo da seção com mais clareza.

Ao realizar um exemplar da versão preliminar do GGDD, disponível no Apêndice Versão Instanciada - Overcooked 2, foi verificado que a subseção **Elementos do enredo**, parte do subtópico **Narrativa** que compõe a seção de **História**, estava agregando informações redundantes, presentes em outras áreas do documento. Desse modo, tal subseção foi retirada.

O item **Introdução** dos tópicos de níveis e das áreas do jogo também foi retirado da seção **Níveis e/ou Mundo do jogo**, já que apresentariam apenas informações redundantes com aquelas presentes em outros tópicos dessa mesma seção.

Na versão inicial do GGDD, não havia o tópico para mapas de áreas específicas do jogo, sendo restrito apenas aos níveis ou a um mapa geral do jogo. Porém, foi constatado que existem jogos que não se utilizam do sistema de níveis, ao mesmo tempo que possuem uma mapa geral muito extenso para ser exibido em uma única parte. Assim sendo, a adição da possibilidade de apresentar o mapa do jogo, dividido em diversas áreas, é uma adição de grande valor ao documento.

Outra alteração realizada encontra-se na seção de **Cronograma**, na qual foi reti-

rada a data de conclusão do *Game Design*, pois se tratando da etapa de *Game Design*, essa pode ser realizada várias vezes durante o processo de desenvolvimento de um jogo, tornando-a volátil e complicada de se estimar, o que poderia não gerar impacto positivo no valor dessa informação para a seção.

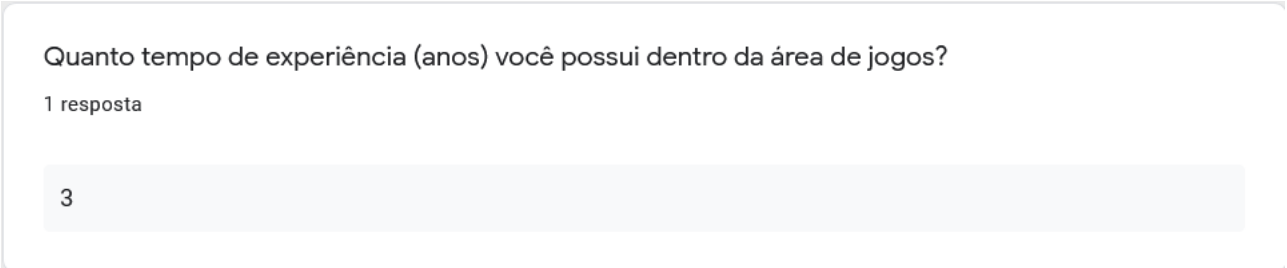
5.3 Segundo ciclo - Avaliação por Especialista

Após realizar os ajustes notados na autoavaliação, foi elaborado o plano de ação desse ciclo. Esse plano consistia no envio do modelo com a estrutura, uma versão instanciada e um questionário a um especialista formado em Engenharia de *Software*, que já trabalhou como professor substituto na **Universidade de Brasília**, possuindo 3 anos de experiência dentro da área de jogos, sendo 2 anos e 10 meses atuando na *Wildlife Studios*, estúdio *indie* brasileiro renomado. O questionário enviado visou capturar o *feedback*/avaliação acerca da proposta, além de sugestões de melhorias. Foi de grande valia ao trabalho, pois se trata de uma visão relacionada ao mercado de jogos. Porém, como só foi obtida uma avaliação, não se pode presumir que seja uma opinião geral.

O questionário foi projetado com um total de três seções. A primeira pretendeu compreender sobre a função e o tempo de experiência do avaliador, além de capturar suas opiniões acerca das práticas de Engenharia de Requisitos inseridas no modelo. A segunda estava condicionada à uma resposta positiva sobre a adição das técnicas gerarem limitações no processo criativo. Ou seja, caso fosse respondido que alguma das técnicas limita o processo criativo, seria indagado qual é a técnica, e por que ela limita essa criatividade. Por fim, a última seção buscou registrar questões a respeito da aplicabilidade da proposta em projetos, ademais de coletar sugestões e melhorias.

As duas perguntas iniciais, representadas pelas Figuras 12 e 13, visavam capturar o perfil do especialista em termos de anos de experiência na área de jogos e a função que desempenha.

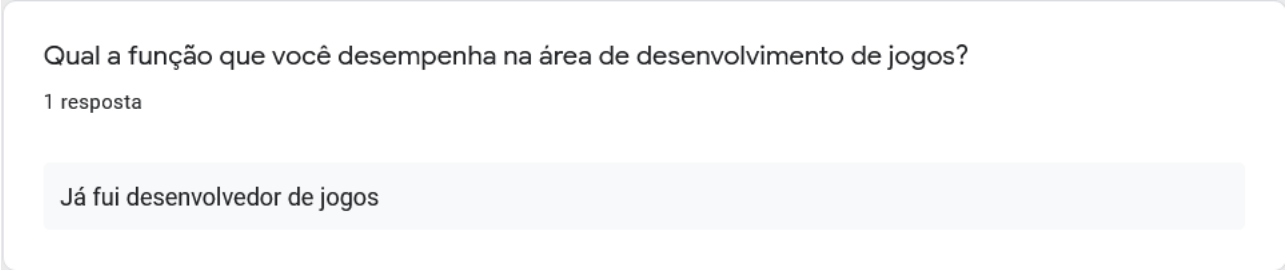
Figura 12 – Tempo de Experiência



Fonte: Autores

Em seguida, foi feito um questionamento sobre quais das práticas referentes à Engenharia de Requisitos, aplicadas ao GGDD, o especialista já conhecia. Tinha-se como

Figura 13 – Função Exercida



Fonte: Autores

objetivo, com essa pergunta, analisar a familiaridade desse especialista com essa área e perceber como tal fator poderia influenciar em sua avaliação. Dessa maneira, foi obtida uma resposta que dentre todos os artefatos e técnicas listadas (Versionamento, *Brainstorming*, Introspecção, *Rich Picture*, Prototipação, Aspectos de Qualidade e MoSCoW), não havia familiaridade apenas com a introspecção e o MoSCoW.

Posteriormente, foi feita a avaliação de cada uma das técnicas, visando obter conhecimento acerca da contribuição, de forma individual, da aplicação de tais técnicas no processo de desenvolvimento de jogos. A avaliação foi realizada através de níveis de contribuição, ou seja, quanto maior a contribuição do artefato ou técnica para o GGDD, maior o seu nível, variando de 1 (nível mais baixo) a 5 (nível mais alto).

O primeiro desses artefatos e técnicas avaliadas foi o versionamento. Segundo o especialista, a aplicação dessa técnica não teria uma contribuição significativa (nível 1) para manter o histórico e controlar as alterações.

A aplicação do *Brainstorming*, visando fazer o registro das ideias envolvidas no processo de desenvolvimento de jogos, obteve uma contribuição extremamente significativa, atingindo o nível máximo (nível 5).

A utilização da Introspecção, técnica introduzida para contribuir no registro de ideias juntamente com o *Brainstorming*, também recebeu, apesar da pouca familiaridade do especialista com a técnica, avaliação positiva, atingindo o nível máximo de contribuição.

O *Rich Picture* foi incorporado à proposta pretendendo auxiliar na compreensão do contexto geral e das principais ideias do jogo de forma visual. Em termos de contribuição, sua avaliação alcançou o nível 4, que está acima do valor mediano (nível 3), possibilitando a consideração de que sua inserção foi positiva.

Em relação à seção de Aspectos de Qualidade, referentes ao NFR *Framework*, o especialista a considerou bastante importante, atribuindo nível 4 em sua avaliação.

O MoSCoW, técnica que o avaliador não possuía grande familiaridade e cuja inclusão desejava fornecer amparo para uma melhor organização e priorização do cronograma relativo ao processo de desenvolvimento do jogo, conseguiu o nível máximo de contribui-

ção.

Em relação à prototipação, foi perguntado em quais desses aspectos a técnica de prototipação poderia ser útil: detalhamento das principais mecânicas, elaboração de mapas e definição de HUDs e menus. Na opinião do especialista, essa técnica poderia afetar positivamente os três aspectos citados na pergunta.

A questão que encerra a primeira seção do questionário trata da limitação em relação ao processo criativo que a adição dessas técnicas poderia provocar. O avaliador acredita que a inserção das técnicas não causaria impacto no processo criativo. Caso a resposta indicasse que alguma das técnicas poderia ter esse impacto, o questionário seria redirecionado para uma pergunta pedindo que o avaliador indicasse quais técnicas e por que elas limitariam o processo criativo. Como isso não ocorreu, não houveram respostas para essa pergunta.

Um dos objetivos do GGDD é facilitar o processo de transição entre o *design* e o desenvolvimento do jogo. Assim, foi questionado ao avaliador se o GGDD, em sua visão, cumpria o objetivo proposto, e foi obtida uma avaliação positiva.

Outro aspecto importante para essa pesquisa é acerca da utilização desse modelo no mercado de jogos. Logo, foi coletado esse *feedback* através do questionamento ("Você usaria esse modelo em algum projeto que fizesse parte?"), no qual foi obtida uma resposta moderada (Talvez), indicando que há possibilidade de seu uso, apesar de não ser uma confirmação.

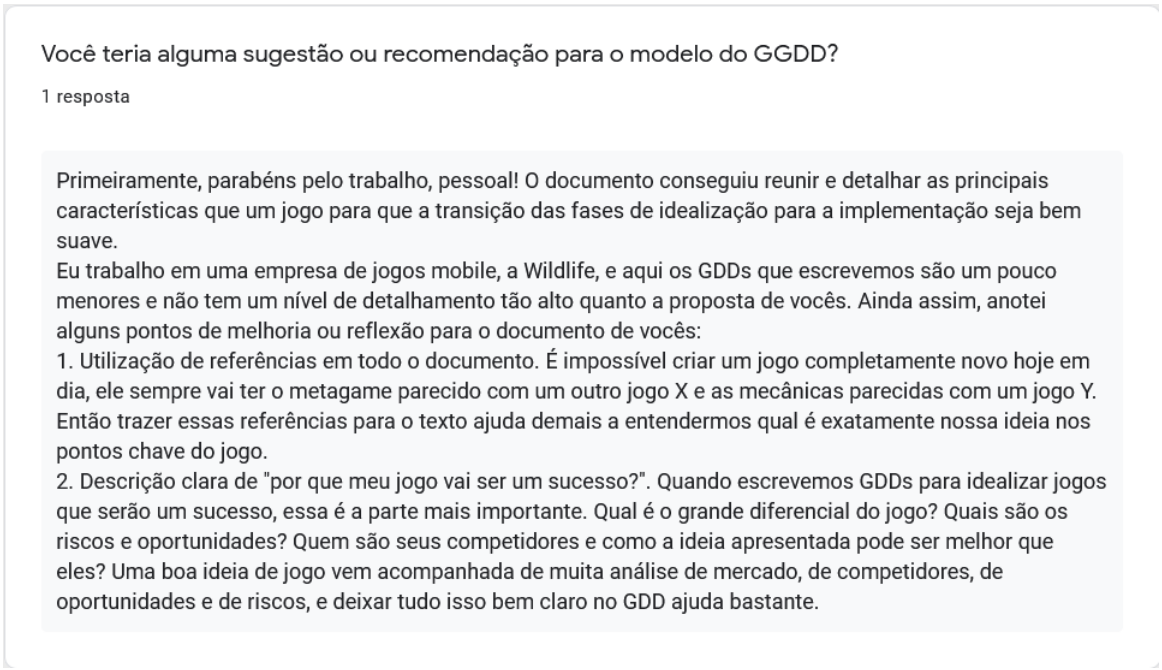
Por fim, foi perguntado se o especialista possuía sugestões para o modelo do GGDD e a resposta está especificada na Figura 14.

5.3.1 Melhorias aplicadas

Após a avaliação feita pelo especialista, as respostas foram analisadas e algumas mudanças foram realizadas no modelo vigente, após o primeiro ciclo. A primeira delas é referente ao versionamento. Em conformidade com o resultado apresentado, esse versionamento não obteve níveis significativos de contribuição. Assim, os autores optaram por indicar, na estrutura do modelo, que deixar de utilizar essa seção não causa problemas muito graves para o desenvolvimento do documento. Ou seja, a utilização dessa seção é recomendada para quem deseja manter um rastro das alterações, mas não é determinante para que se tenha um bom GGDD.

Tendo em vista as sugestões recomendadas pelo avaliador, ocorreu a inclusão de uma seção para que se pudesse avaliar os fatores competitivos relativos ao jogo. No início da seção, encontra-se uma matriz SWOT, visando analisar as forças, fraquezas, oportunidades e ameaças do jogo. No final, há um tópico para comparação com o mercado.

Figura 14 – Sugestões



Fonte: Autores

Tendo em vista as sugestões recomendadas pelo avaliador, foi incluída uma seção, à parte, e ao final do documento, para registro das referências de jogos utilizadas quando o usuário elaborar seu GGDD, pois sempre que alguma característica, desde uma pequena mecânica ou todo o estilo do gráfico, tiver como base um jogo já existente, deve-se colocar a referência para esse jogo na parte do documento onde essa característica foi citada, e explicitar esse jogo base nessa seção.

5.4 Terceiro ciclo - Comparação com proposta similar

Durante o estudo da literatura acerca do tema em pesquisa, foi encontrado um modelo de GDD com proposta semelhante. Esse modelo, denominado iGDD, foi elaborado por (SALAZAR et al., 2012). Com o objetivo de entender em que aspectos os GDDs são semelhantes, foi elaborado um plano de ação que envolve a comparação da estrutura e dos objetivos de ambos os modelos. Inicialmente, foi construída a Tabela 4, que apresenta um compilado das seções do GGDD, e relaciona com as seções de propósitos semelhantes do iGDD. Em seguida, é feita uma comparação mais aprofundada dos modelos.

O primeiro fator observado, ao comparar ambas estruturas, está relacionado ao **Histórico de Versão**. No GGDD, é uma seção opcional, porém sugerida visando auxiliar na rastreabilidade do projeto. A não utilização dessa seção foi a opção escolhida pelos autores do iGDD.

A seção de *Brainstorming* e Introspecção foi criada especificamente para acoplar

Tabela 4 – Comparação de Estruturas

GGDD	**iGDD por (SALAZAR et al., 2012)**
Histórico de Versão	—
Brainstorming e Introspecção	—
Visão Geral do Jogo	*Overview*
Fatores Competitivos	*Overview - Game Justification*
Gameplay e Mecânicas Principais	*Mechanics*
Aspectos de Qualidade	*Experience*
História	*Dynamics - Game Detailed History*
Níveis e/ou mundo do jogo	*Dynamics*
Interface	*Aesthetics - Game Interface Visual*
Arte e vídeo	*Aesthetics*
Música e efeitos sonoros	*Aesthetics*
Sistemas Alvo	*Constraints and Assumptions - Technical Constraints*
Cronograma	—
Referências	*Document Information*

essas duas técnicas ao documento. Dessa forma, como era esperado, não há nenhuma seção relacionada a isso no iGDD.

A terceira e última seção do GGDD, que não possui correspondência, é o cronograma, adicionado, principalmente, para o uso da técnica MoSCoW, para auxiliar na organização do processo.

Com exceção das três citadas acima, todas as outras seções do GGDD possuem tópicos semelhantes no iGDD, mesmo que, em alguns casos, uma seção inteira do primeiro esteja relacionada a apenas um pequeno tópico do segundo e vice-versa. Um bom exemplo disso são as questões relacionadas a um possível sucesso do jogo. Enquanto o GGDD dedica uma seção inteira para isso (Fatores competitivos), inclusive utilizando-se de técnicas como a matriz SWOT, o iGDD apenas cita um tópico para a justificativa do jogo e possíveis razões para seu sucesso.

Apesar das diferenças, ambos os modelos buscam analisar a parte da experiência do usuário com o jogo. Isso fica evidenciado pela existência de um capítulo para a experiência do usuário no modelo de (SALAZAR et al., 2012), e para os aspectos de qualidade no modelo de GGDD.

A forma de abordagem também é realizada de maneira distinta. Salazar et al. (2012) utilizam uma estrutura de GDD com elementos em comum no processo de *design* de um jogo encontrados durante sua pesquisa, incorporando, associada a essa estrutura, práticas recomendadas para uma boa *Software Requirements Specification* (SRS) sugeridas pela IEEE. Já o GGDD é mais voltado para a introdução de artefatos e técnicas da Engenharia de Requisitos ao GDD. Apesar disso, o propósito de ambos é facilitar a transição entre as diferentes fases de um processo de desenvolvimento do jogo, consequentemente,

diminuindo a quantidade necessária de retrabalho ao longo de todo esse processo.

Em resumo, ao se fazer uma análise geral do modelo de GGDD e do proposto por (SALAZAR et al., 2012), foi possível perceber que ambas as propostas possuem objetivos semelhantes, apesar das visíveis diferenças na estrutura. Os dois modelos buscam, ao mesmo tempo, abranger as necessidades técnicas do jogo, mas sem esquecer do sentimento dos jogadores ao jogar esse jogo.

5.5 Resumo do Capítulo

O presente capítulo demonstrou os resultados obtidos ao longo dos três ciclos de pesquisa ação realizados. O primeiro desses ciclos é uma autocrítica, realizada pelos próprios autores, com base no modelo preliminar do GGDD. O segundo baseou-se em uma análise do modelo feita por um especialista na área. Já no terceiro, foi realizada uma comparação entre o modelo proposto do GGDD e o iGDD, modelo elaborado por (SALAZAR et al., 2012).

Os dois primeiros ciclos geraram alterações no modelo inicialmente proposto, deixando-o mais robusto, e aumentando a possibilidade de fazê-lo atingir seu propósito. Enquanto isso, o terceiro ciclo busca entender as semelhanças e as diferenças entre duas propostas, as quais procuram atingir objetivos equivalentes.

6 Conclusão

Este capítulo busca apresentar as conclusões acerca do que foi alcançado durante a elaboração deste TCC. Serão retomados os objetivos apresentados no capítulo de Introdução, visando analisar se o que foi proposto conseguiu ser efetivamente alcançado. Em seguida, serão ressaltados os pontos fortes e as possíveis fragilidades da proposta apresentada. Por fim, são explicitadas possíveis contribuições futuras para o modelo de GGDD e o *site*.

6.1 Objetivos alcançados

O primeiro objetivo foi definir o modelo de GDD que servisse de base para a proposta. Esse objetivo foi alcançado com a elaboração do modelo preliminar, apresentado no seguinte Apêndice Modelo Preliminar - Versão TCC1.

O objetivo seguinte era realizar a definição das práticas da Engenharia de Requisitos que poderiam ser úteis no contexto de desenvolvimento de jogos. Esse objetivo foi alcançado, e as técnicas e os artefatos selecionados encontram-se no modelo proposto, apresentado no capítulo sobre o GGDD.

Após isso, foi proposto o envio do modelo para especialistas da área de jogos e a realização de uma análise com base em suas respostas. Essa proposição foi cumprida, e está registrada na seção Segundo ciclo - Avaliação por Especialista do capítulo de Análise de Resultados.

Outra meta era comparar o modelo proposto com o sugerido por Salazar et al. (2012). Esse aspecto está apresentado na seção Terceiro ciclo - Comparação com proposta similar, presente no capítulo de Análise de Resultados.

Por último, foi estabelecido o desenvolvimento de um *website* explicativo com base no modelo levantado. Esse produto foi elaborado, sendo disponibilizado no seguinte *link*, com maiores informações encontradas em *Site* Informativo.

Diante do exposto, pode-se responder a questão de pesquisa: "É possível obter um modelo de GDD, orientado às boas práticas da Engenharia de Requisitos, permitindo aos desenvolvedores de jogos uma compreensão mais adequada quanto às ideias arquitetadas pelos *designers*, durante a fase de pré-produção, a fim de aplicá-las corretamente ao longo das demais etapas do desenvolvimento do jogo?"

Como resposta, pode-se dizer "sim" para a obtenção de um modelo de GDD, orientando-se por boas práticas da Engenharia de Requisitos. Adicionalmente, tem-se a

opinião de um especialista, com três anos de mercado em uma empresa de renome da área de jogos, que atestou positivamente para uma compreensão mais adequada quanto às ideias, inerentes da fase de pré-produção. Entretanto, trata-se de uma opinião, sendo necessárias outras consultas bem como de uma análise mais aprofundada para se obter um resultado mais conclusivo quanto a essa parte da questão de pesquisa.

6.2 Competências do GGDD

O GGDD é um modelo que preza pelo lado criativo e ao mesmo tempo visa inserir técnicas advindas da Engenharia de Requisitos de extrema valia ao processo de desenvolvimento de jogos. Entende-se que essas técnicas auxiliam a transição da pré-produção para a produção, momento crítico nesse processo.

O uso dessas técnicas e desses artefatos aumenta a quantidade de possibilidades na elaboração do documento, e confere aos responsáveis pelo *design* do jogo maior versatilidade para esse processo.

Foi elaborado um documento com a estrutura e uma explicação de todas as seções para consulta, além de uma versão instanciada dessa estrutura para sanar qualquer dúvida a respeito do documento.

Por fim, foi desenvolvido *site* informativo para o modelo que, além de disponibilizar as versões previamente citadas para *download*, contém a explicação da proposta e maiores informações acerca dessa pesquisa.

6.3 Fragilidades do GGDD

A principal fragilidade encontrada durante essa pesquisa refere-se ao tamanho do GGDD. Assim como apontado pelo especialista, durante o Segundo ciclo - Avaliação por Especialista da Análise de Resultados, algumas das equipes de desenvolvimento optam por GDDs mais enxutos, para reduzir o tempo de documentação e acelerar o processo. Dessa forma, uma proposta com o tamanho do GGDD pode desmotivar o seu uso por parte de algumas equipes.

Uma outra fragilidade refere-se à análise por parte do especialista. Inicialmente, a ideia era consultar vários especialistas diferentes. Porém, devido às dificuldades no contato com esses especialistas e aos prazos justos, apenas um desses especialistas respondeu a pesquisa. Apesar de bem conceituado, sua opinião e forma de trabalho não representam o pensamento de todos da área. Ou seja, ainda que sua análise sobre o GGDD tenha sido positiva, o modelo pode não agradar a todos no contexto de desenvolvimento de jogos.

Adicionalmente, cabe colocar que, apesar da literatura especializada acordar sobre

a relevância da Engenharia de Requisitos na especificação de um software, em especial de alta complexidade como é o caso dos jogos, o mercado tem alta resistência quanto ao uso de práticas associadas à área. Portanto, como o GGDD acorda o uso dessas práticas, pode-se ter certa resistência por parte de equipes já acostumadas com o modo operando do mercado de jogos. Essa fragilidade, na verdade, não está associada ao GGDD em si, mas à cultura inerente do mercado, e de como eles conduzem o desenvolvimento de software.

6.4 Trabalhos Futuros

Em termos de trabalhos futuros, foram identificadas as seguintes possibilidades: (i) deixar o modelo mais sucinto, sem afetar o propósito original da proposta; (ii) criar uma ferramenta que pode ser integrada ao *site* para elaboração colaborativa do modelo; (iii) evoluir a opção de *download* via formulário do *site*, adicionando mais trechos do GGDD, além de inserir as respostas do formulário nas suas respectivas seções no documento, e (iv) criar uma ferramenta que, a partir do código fonte do jogo, gere trechos do GGDD, para auxiliar na manutenibilidade do documento.

Referências

ABREU, F. B. E.; GOULãO, M. Coupling and cohesion as modularization drivers: are we being over-persuaded? *Proceedings Fifth European Conference on Software Maintenance and Reengineering*, IEEE, p. 47–57, 2001. Citado na página 27.

ALEEM, S.; CAPRETZ, L. F.; AHMED, F. Critical Success Factors to Improve the Game Development Process from a Developer's Perspective. *Journal of Computer Science and Technology*, Springer Science + Business Media, LLC Science Press, China, p. 925–948, 2016. Citado 6 vezes nas páginas 19, 30, 34, 35, 36 e 42.

ALEEM, S.; CAPRETZ, L. F.; AHMED, F. Game development software engineering process life cycle: a systematic review. *Journal of Software Engineering Research and Development*, Springer, p. 1–30, 2016. Citado 2 vezes nas páginas 34 e 36.

ALI, A. et al. Role of Requirement Prioritization Technique to Improve the Quality of Highly-Configurable Systems. *IEEE Access*, IEEE, p. 27549–27573, 2020. Citado na página 27.

BALDWIN, M. *Game Design Document Outline*. 2005. Citado 2 vezes nas páginas 49 e 50.

BECK, K.; ANDRES, C. *Extreme Programming Explained: Embrace Change*. 2nd. ed. Boston, EUA: Addison-Wesley Professional, 2004. ISBN 978-0-321-27865-4. Citado na página 29.

BETHKE, E. *Game Development and Production*. Texas, EUA: Wordware Publishing, Inc., 2002. ISBN 0-71246-321-3. Citado 6 vezes nas páginas 31, 34, 35, 36, 42 e 49.

BLOW, J. Game Development: Harder Than You Think: Ten or twenty years ago it was all fun and games. Now it's blood, sweat, and code. *Queue*, ACM, p. 29–37, 2004. Citado na página 30.

BROWN, D. M. *Communicating design: Developing web site documentation for design and planning*. California, EUA: [s.n.], 2010. ISBN 1-55622-951-8. Citado na página 35.

CALLELE, D.; NEUFELD, E.; SCHNEIDER, K. Requirements Engineering and the Creative Process in the Video Game Industry. *Proceedings of the 2005 13th IEEE International Conference on Requirements Engineering (RE'05)*, IEEE, Saskatoon, Canadá, 2005. Citado 3 vezes nas páginas 18, 33 e 42.

CALLELE, D.; NEUFELD, E.; SCHNEIDER, K. A Report on Select Research Opportunities in Requirements Engineering for Videogame Development. *Proceedings of the 2011 Fourth International Workshop on Multimedia and Enjoyable Requirements Engineering (MERE'11)*, IEEE, Saskatchewan, Canadá, 2011. Citado 3 vezes nas páginas 19, 34 e 35.

CALLELE, D.; WNUK, K.; PENZENSTADLER, B. New Frontiers for Requirements Engineering. *2017 IEEE 25th International Requirements Engineering Conference*, IEEE, 2017. Citado na página 24.

COLBY, R.; COLBY, R. S. Game Design Documentation: Four Perspectives from Independent Game Studios. *Communication Design Quartely*, ACM, Denver, EUA, p. 1–11, 2019. Citado 2 vezes nas páginas 34 e 35.

COMMITTEE, S. Engineering Standards. Ieee Recommended Practice for Software Requirements Specifications. IEEE, 1998. Citado na página 27.

COOKE, O. J. *Silent Hill 2 Design Document*. 2001. <https://drive.google.com/file/d/1nxvdXasP-HsRCt62cHK3wF_pIrJpYx5T/view>. Acesso em 30/04/2021. Citado 2 vezes nas páginas 49 e 51.

DANEVA, M. How Practitioners Approach Gameplay Requirements? An Exploration into the Context of Massive Multiplayer Online Role-Playing Games. *RE 2014, Karlskrona, Suécia*, IEEE, Enschede, Holanda, 2014. Citado na página 19.

DMA Design Ltd. *Race'n'Chase Game Design*. 1995. <https://www.gamedevs.org/uploads/grand-theft-auto.pdf>. Acesso em 30/04/2021. Citado 2 vezes nas páginas 49 e 51.

ENGEL, G. I. Pesquisa-ação. *Educar n. 16*, UFPR, Curitiba, Brasil, p. 181–191, 2000. Citado na página 43.

ENGSTRöM, H. et al. Game development from a software and creative product perspective: A quantitative literature review approach. *Entertainment Computing 27*, Elsevier B.V, Skövde, Suécia, p. 10–22, 2018. Citado 3 vezes nas páginas 17, 30 e 31.

ESPINHA, R. G. *Kanban*. 2019. <https://artia.com/kanban/>. Acesso em 02/04/2021. Citado na página 29.

FAGAN, M. E. Design and Code Inspections to Reduce Errors in Program Development. *Pioneers and Their Contributions to Software Engineering*, Springer, p. 301–334, 2001. Citado na página 28.

FATIMA, A.; RASOOL, T.; QAMAR, U. GDGSE: Game Development with Global Software Engineering. *2018 IEEE Games, Entertainment, Media Conference (GEM)*, IEEE, Islamabad, Paquistão, 2018. Citado 2 vezes nas páginas 19 e 30.

FIGMA. *Figma: the collaborative interface design tool*. 2021. <https://www.figma.com/>. Acesso em 21/03/2021. Citado na página 37.

FIGMA. *Prototyping*. 2021. <https://www.figma.com/prototyping/>. Acesso em 21/03/2021. Citado na página 37.

GERHARDT, T. E.; SILVEIRA, D. T. *Métodos de Pesquisa*. 7th. ed. Porto Alegre, Brasil: UFRGS, 2009. ISBN 978-85-386-0071-8. Citado na página 40.

GIL, A. C. *Métodos e Técnicas de Pesquisa Social*. 6th. ed. São Paulo, Brasil: Editora Atlas S.A., 2008. ISBN 978-85-224-5142-5. Citado 3 vezes nas páginas 40, 41 e 43.

GOGUEN, J. A.; LINDE, C. Techniques for requirements elicitation. *Proceedings, Requirements Engineering '93*, IEEE, San Diego, EUA, p. 152–164, 1993. Citado na página 26.

GONZáLEZ-SALAZAR, M.; MITRE-HERNáNDEZ, H.; LARA-ALVAREZ, C. Method for Game Development Driven by User-experience: A Study of Rework, Productivity and Complexity of Use. *International Journal of Advanced Computer Science and Applications, Vol. 8, No. 2*, IJACSA, Zacatecas, México, 2017. Citado na página 19.

Heroku. *What is Heroku?* 2021. <https://www.heroku.com/what>. Acesso em 23/10/2021. Citado na página 39.

KANODE, C. M.; HADDAD, H. M. Software Engineering Challenges in Game Development. *2009 Sixth International Conference on Information Technology: New Generations*, IEEE, Kennesaw, EUA, p. 260–265, 2009. Citado 7 vezes nas páginas 17, 18, 31, 32, 33, 36 e 42.

KOUTONEN, J.; LEPPäNEN, M. How are Agile Methods and Practices Deployed in Video Game Development? a Survey into Finnish Game Studios. *Agile Processes in Software Engineering and Extreme Programming*, Springer, Jyväskylä, Finlândia, p. 135–149, 2013. Citado na página 31.

K.SCHWABER; SUTHERLAND, J. *The Scrum guide – The definitive guide to Scrum: The rules of the game*. [S.l.: s.n.], 2011. Citado na página 31.

LEITE, J. C. S. P. Engenharia de Requisitos Notas de Aula, Parte ii. Rio de Janeiro, Brasil, 1994. Citado na página 24.

LEWIS, C.; WHITEHEAD, J. The Whats and the Whys of Games and Software Engineering. *GAS' 11*, ACM, Califórnia, EUA, p. 1–4, 2011. Citado na página 30.

MDN WEB DOCS. *CSS*. 2021. <https://developer.mozilla.org/pt-BR/docs/Web/CSS>. Acesso em 21/03/2021. Citado na página 38.

MDN WEB DOCS. *HTML: Linguagem de Marcação de Hipertexto*. 2021. <https://developer.mozilla.org/pt-BR/docs/Web/HTML>. Acesso em 21/03/2021. Citado na página 37.

MDN WEB DOCS. *HTML5*. 2021. <https://developer.mozilla.org/pt-BR/docs/Web/Guide/HTML/HTML5#introdu%C3%A7%C3%A3o_ao_html5>. Acesso em 21/03/2021. Citado na página 37.

MDN WEB DOCS. *Sobre JavaScript*. 2021. <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/About_JavaScript>. Acesso em 21/03/2021. Citado na página 38.

MITRE-HERNáNDEZ, H. A. et al. Decreasing Rework in Video Games Development from a Software Engineering Perspective. *Advances in Intelligent Systems and Computing 405*, Springer International Publishing Switzerland, Zacatecas, Mexico, 2016. Citado na página 31.

MONK, A.; HOWARD, S. Methods tools: The rich picture: A tool for reasoning about work context. *Interactions*, Association for Computing Machinery, Nova Iorque, EUA, p. 21–30, 1998. Citado 2 vezes nas páginas 25 e 26.

MOWAD ramdan; MUHAMMAD, K. A.; ELFAKHARANY, E. A Supporting Tool for Requirements Prioritization Process in Agile Software Development. *Future Computing and Informatics Journal*, Cairo, Egito, 2020. Citado na página 27.

MURPHY-HILL, E.; ZIMMERMANN, T.; NAGAPPAN, N. Cowboys, Ankle Sprains, and Keepers of Quality: How Is Video Game Development Different from Software Development? *ICSE' 14*, ACM, EUA, p. 1–11, 2014. Citado na página 33.

MUSIL, J. et al. Improving Video Game Development: Facilitating Heterogeneous Team Collaboration Through Flexible Software Processes. *Communications in Computer and Information Science*, Springer, Vienna, Áustria, p. 83–94, 2010. Citado 2 vezes nas páginas 30 e 32.

NIELSEN, J. *Usability Engineering*. [S.l.]: Academic Press, Inc., 1993. ISBN 0-12-518406-9. Citado na página 24.

NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: A roadmap. *Proceedings of the Conference on The Future of Software Engineering*, Association for Computing Machinery, Nova Iorque, EUA, p. 35–46, 2000. Citado 2 vezes nas páginas 25 e 26.

OLIVEIRA, P. H. R. L. de; MIRANDA, C. A. S. de; GOMIDE, J. V. B. Game design tools for maximum effectiveness. *International Conferences Interfaces and Human Computer Interaction 2018; Game and Entertainment Technologies 2018; and Computer Graphics, Visualization, Computer Vision and Image Processing 2018*, IADIS, Belo Horizonte, Brasil, p. 363–367, 2018. Citado na página 53.

PERFORCE. *HelixRM: The Requirements Management Tool for Higher Product Quality*. 2021. <https://www.perforce.com/products/helix-requirements-management?r=rmt&dpm=37469>. Acesso em 23/03/2021. Citado na página 28.

PETRILLO, F.; PIMENTA, M. Is agility out there? Agile practices in game development. *SIGDOC 2010*, ACM, Porto Alegre, Brasil, p. 9–15, 2010. Citado 2 vezes nas páginas 31 e 32.

PETRILLO, F. et al. Houston, we have a problem...: A Survey of Actual Problems in Computer Games Development. *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, ACM, Porto Alegre, Brasil, p. 707–711, 2008. Citado 3 vezes nas páginas 17, 19 e 33.

PRESSMAN, R. S. *Engenharia de Software: Uma Abordagem Profissional*. 7th. ed. Porto Alegre, Brasil: AMGH Editora Ltda, 2011. ISBN 978-85-8055-044-3. Citado na página 22.

ROUNGAS, B. A Model-driven Framework for Educational Game Design. *International Journal of Serious Games*, DOAJ, Delft, Holanda, p. 19–37, 2016. Citado na página 35.

SALAZAR, M. G. et al. Proposal of Game Design Document from Software Engineering Requirements Perspective. *The 17th International Conference on Computer Games*, IEEE, p. 81–85, 2012. Citado 8 vezes nas páginas 17, 19, 20, 42, 66, 67, 68 e 69.

SAYãO, M.; LEITE, J. Rastreabilidade de Requisitos. *RITA*, p. 57–86, 2006. Citado na página 28.

SCHWABER, K.; SUTHERLAND, J. *The 2020 Scrum Guide™*. 2020. <https://scrumguides.org/scrum-guide.html>. Acesso em 02/04/2021. Citado na página 29.

Silva LL. Análise SWOT. 2009. Citado na página 23.

SOMMERVILLE, I. *Engenharia de Software*. 9th. ed. São Paulo, Brasil: Pearson Education, Inc., 2011. ISBN 978-85-7936-108-1. Citado 5 vezes nas páginas 22, 23, 24, 25 e 26.

SOUZA, L. P. S. e et al. Matriz swot como ferramenta de gestÃo para melhoria daassistÊncia de enfermagem: Estudo de caso em um hospital deensino. *Revista Eletrônica Gestão Saúde*, p. 1633–1643, 2013. Citado na página 24.

TAYLOR, C. *Game Design Document Template*. 1999. Citado 2 vezes nas páginas 49 e 50.

THIOLLENT, M. *Metodologia da pesquisa-ação*. 14th. ed. São Paulo, Brasil: Cortez Editora, 2005. Citado na página 43.

WALFISZ, M.; ZACKARIASSO, P.; WILSON, T. L. Real-time strategy: Evolutionary game development. *Business Horizons*, Elsevier, Indiana, EUA, p. 487–498, 2006. Citado na página 35.

WANG, A. I.; NORDMARK, N. Software Architectures and the Creative Processes in Game Development. *14th International Conference on Entertainment Computing (ICEC)*, Trondheim, Noruega, p. 272–285, 2015. Citado 2 vezes nas páginas 20 e 30.

WATERS, K. *Prioritization using MoSCoW*. 2009. <https://cs.anu.edu.au/courses/comp3120/local_docs/readings/Prioritization_using_MoSCoW_AllAboutAgile.pdf>. Acesso em 22/03/2021. Citado na página 27.

WIEGERS, K. First things first: prioritizing requirements. *Software Development*, p. 48–53, 1999. Citado na página 27.

YOUNG, R. R. *The Requirements Engineering Handbook*. Massachusetts, EUA: ARTECH HOUSE, INC., 2004. ISBN 1-58053-266-7. Citado 2 vezes nas páginas 25 e 28.

YRJöNEN, A.; MERILINNA, J. Extending the nfr framework with measurable non-functional requirements. 2009. Citado na página 26.

ZAVE, P. Classification of Research Efforts in Requirements Engineering. *ACM Computing Surveys, Vol. 29, No. 4*, ACM, EUA, 1997. Citado na página 24.

ZENHUB. *ZenHub Browser Extension*. 2021. <https://www.zenhub.com/extension>. Acesso em 27/03/2021. Citado na página 38.

ZYDA, M. From Visual Simulation to Virtual Reality to Games. *Computer*, IEEE, p. 25–32, 2005. Citado na página 32.

Apêndices

APÊNDICE A – Modelo Preliminar - Versão TCC1

- Capa
  - Nome do jogo
  - Autor
  - Ano
  - Versão
  - Informações de *copyright*
- Histórico de versão
- Sumário
- Visão geral do jogo
  - Descrição geral do jogo
  - Descrição do ambiente/clima do jogo
  - Gênero
  - Público-alvo
  - Escopo do projeto
    - Quantidade de localidades
    - Quantidade de níveis
    - Quantidade de inimigos
    - Quantidade de NPCs (Personagens não jogáveis)
    - ...
- *Gameplay* e mecânicas principais
  - *Gameplay*
    - Visão do jogo
    - Objetivos
    - Progressão no jogo
  - Mecânicas
    - Física

        - Ações do personagem principal
        - Combate
            - Inimigos
            - Modelagem do combate
        - Sistema de economia
        - *Inputs* recebidos

- História
    - Narrativa
        - *Background* da história
        - Elementos do enredo
        - Encerramento da história
            - Múltiplos finais
    - Personagens
        - Personagem 1
            - Nome e história
            - Personalidade
            - Características
            - Papel na narrativa
            - Relação com os outros personagens
        - Personagem 2
    - *Cutscenes*
        - *Cutscene* 1
            - Personagens
            - Descrição
            - *Script*
        - *Cutscene* 2

- Níveis e/ou mundo do jogo (de acordo com a abordagem do jogo)
    - Níveis
        - Nível 1
            - Resumo
            - Introdução
            - Mapa
            - Caminho principal

            - Caminho secundário
            - Conclusão
        - Nível 2
    - Mundo do jogo
        - Visão geral do mundo
        - Área 1
            - Descrição geral
            - Características específicas
            - Conexão com outras áreas
            - Níveis (se houver) que utilizam essa área

          Introdução
        - Área 2

- Interface
    - Menu
    - *Head-up displays* (HUDs)
    - Sistemas de ajuda (Tutoriais)
- Arte e vídeo
    - Conceito de arte
    - 2D
    - 3D
    - Cinemáticas
- Música e efeitos sonoros
    - Proposta do *design* de som
    - Músicas
    - Efeitos sonoros
- Sistemas alvo
    - Plataforma 1
        - Requisitos de *hardware*
- Cronograma
    - Data de início do projeto

- Data de conclusão do *game design*
- *Milestone* 1
  - Descrição
  - Objetivo
  - Prazo
- *Milestone* 2
  - Descrição
  - Objetivo
  - Prazo
- Data de conclusão de projeto

Após a definição desse modelo base, práticas e técnicas inerentes à Engenharia de Requisitos, elucidadas no Referencial Teórico, serão acrescidas à essa estrutura. A primeira técnica sugerida é o *Rich Picture*, que está no âmbito da Pré-Rastreabilidade, e seria aplicada afim de conferir uma abordagem mais visual ao tópico de "Visão geral do jogo". Apesar de se tratar de um rascunho inicial, permite uma compreensão da possível extensão e o contexto do projeto. Além de ser de fácil entendimento, é um artefato colaborativo, permitindo a participação de todos os membros da equipe em sua elaboração.

Para auxiliar na elicitação dos requisitos do jogo, as técnicas de *brainstorming* e introspecção possuirão uma seção própria no documento. A primeira é uma técnica simples, colaborativa e, se bem sucedida, acorda um conjunto de boas ideias que auxilia na concepção do jogo. Já a segunda, por envolver uma análise das propriedades necessárias ao jogo, auxilia no levantamento de requisitos para esse. Ainda buscando definir funcionalidades para o jogo, a prototipação será incluída nos tópicos "*Gameplay* e mecânicas principais", "Níveis e/ou mundo do jogo" e "Interface".

A priorização é uma importante questão no desenvolvimento de jogos e, por esse motivo, a técnica MoSCoW se fará presente no cronograma, buscando facilitar o entendimento da equipe em relação às prioridades de desenvolvimento.

O sentimento dos jogadores e as sensações experimentadas por eles ao jogarem um jogo são referências de extrema importância para o desenvolvimento de um jogo. Partindo disso, uma seção à parte, contendo os requisitos não funcionais, estará presente. Nessa seção, serão abordados pontos de qualidade necessários para cumprir esse requisito e estabelecidas formas de operacionalizar esses tópicos. Basicamente, será feita uma adaptação do modelo conceitual proposto pelo *NFR Framework*. Essa adaptação será realizada para evitar uma imposição de uma notação não popular, o que poderia acarretar em rejeições da proposta, além de conferir um certo engessamento, característica indesejável no modelo proposto.

Para manter um rastro de todas as alterações realizadas no documento, a seção "Histórico de versão" armazenará um histórico contendo todas as informações acerca de modificações, explicitadas de maneira detalhada.

APÊNDICE B – Versão Instanciada - Overcooked 2

Ghost Town Games

Overcooked 2

Versão 0.1

02 de Julho, 2021

B.1 Histórico de Versão

B.1.1 Versão 0.1

Breve texto explicativo sobre o foco das mudanças. As mudanças:

- Alteração 1
- Alteração 2
- Alteração 3

B.1.2 Versão x.x

B.2 Sumário

B.3 *Brainstorming* e Introspecção

Aqui devem ser colocadas as ideias levantadas para o desenvolvimento do jogo, incluindo as que foram descartadas, buscando gerar um rastro dos principais conceitos do projeto. Por exemplo:

- Jogo de cozinha e preparação de alimentos
- Alguns alimentos devem ser cozidos, outros fritos e alguns cortados
- Possibilidade de jogar com 1 à 4 jogadores
- Multiplayer local ou online

B.4 Visão Geral do Jogo

B.4.1 Descrição Geral do Jogo

Jogo cooperativo ambientado no Reino da Cebola cujo objetivo é realizar diversas etapas do processo de preparação de pratos em diversos níveis com temáticas e desafios distintos entre si.

B.4.2 Descrição do Ambiente/Clima do Jogo

O Reino da Cebola é um mundo rico, cheio de cozinhas cruéis e estranhas para você conquistar. Participe dessa jornada épica e enfrente cozinhas cada vez mais desafiadoras e bizarras que levarão suas habilidades de cooperação e coordenação ao limite. Cada nível

terá um novo desafio para você e sua equipe superarem, seja andar por aí num navio pirata, passar entre caminhões em alta velocidade, cozinhar no gelo flutuante ou servir comida nas entranhas de um submundo escaldante, cada nível testará a coragem até dos chefes mais valentes.**

** Retirado da página do Overcooked 1 na Epic Games

B.4.3 Gênero

Ação e simulação.

B.4.4 Público-alvo

Família.

B.4.5 Escopo do projeto

B.4.5.1 Quantidade de Localidades

Modo história: 6 localidades no jogo base, com 6 níveis cada.

Modo *arcade*: 19 temas, 9 requerem aquisição de DLCs, com 6 fases por tema.

Modo versus: 19 temas, 9 requerem aquisição de DLCs, com 6 fases por tema.

B.4.5.2 Quantidade de Níveis

Modo história: 130 níveis + 18 níveis bônus

Modo *arcade*: 114 níveis

Modo versus: 114 níveis

B.4.5.3 Quantidade de Inimigos

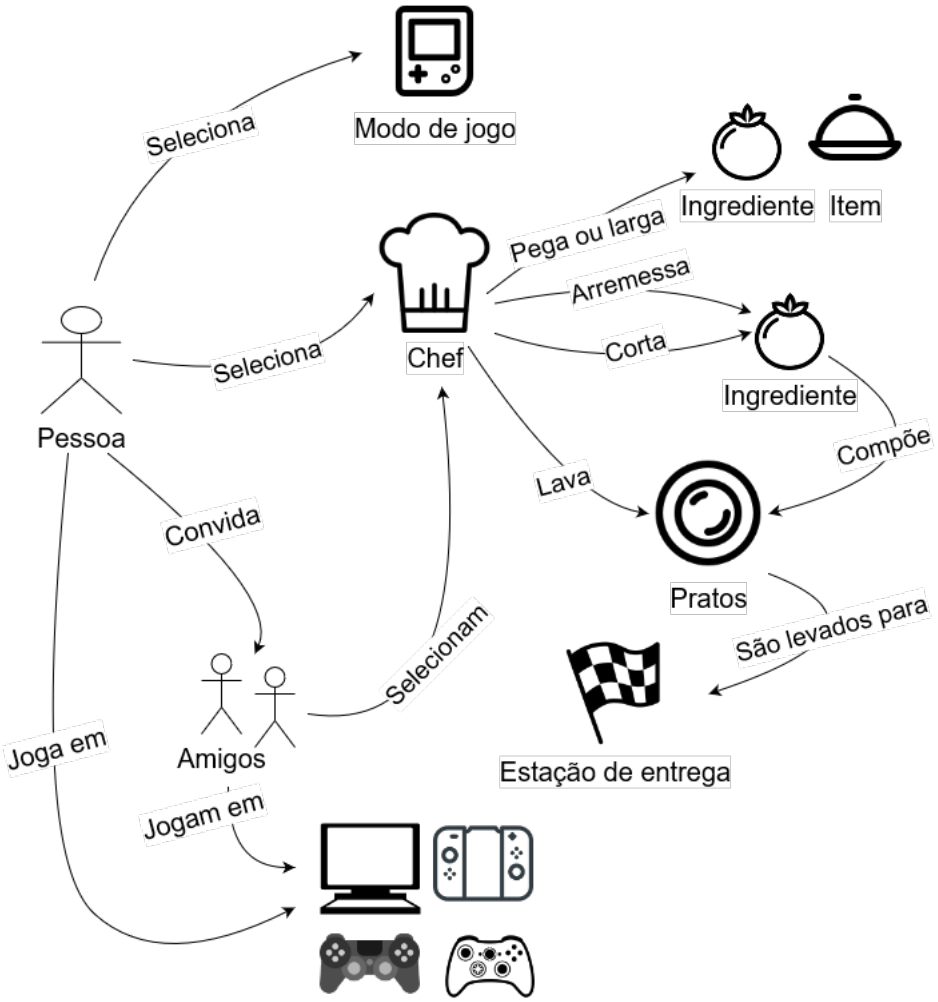
O jogo não terá inimigos em forma de personagens, somente, desafios.

B.4.5.4 Quantidade de NPCs(Personagens não jogáveis)

Rei Cebola, Kevin (Cachorro do Rei Cebola) e os pão-demônios.

B.4.6 **Rich Picture**

Figura 15 – *Rich Picture*



Fonte: Autores

B.5 Fatores Competitivos

B.5.1 **Matriz SWOT**

Figura 16 – *Matriz SWOT*

	Pontos Fortes	Pontos Fracos
Internos	Permite até 4 jogadores; Ter expansões de conteúdo; Possuir vários modos de jogo; Possibilidade de jogo online;	Não possui crossplay entre diferentes plataformas;
Externos	Poucos jogos nessa proposta; Público conquistado com o jogo antecessor;	Risco de rejeição do estilo; Custo dos impostos em alguns países;

Fonte: Autores

B.5.2 **Concorrentes**

Não foram encontrados muitos concorrentes nesse estilo, por isso foi selecionado um jogo de cozinha para uma comparação: Cooking Simulator [1]

Vantagens: Mais simples e rápido de aprender, permite jogo *multiplayer*, mais interessante para quem deseja algo em um nível mais *arcade*.

Desvantagens: Receitas menos elaboradas, menos interessante para quem busca uma proposta ao nível simulação.

B.6 ***Gameplay* e Mecânicas Principais**

B.6.1 ***Gameplay***

B.6.1.1 Objetivos

B.6.1.1.1 Modo História

Alcançar, em cada fase, pontuação suficiente para desbloquear a seguinte e seguir o progresso da história.

B.6.1.1.2 Modo *Arcade*

Completar o objetivo proposto para o nível selecionado.

B.6.1.1.3 Modo Versus

Vencer o oponente em um confronto 1 vs 1.

B.6.1.2 Progressão no Jogo

B.6.1.2.1 Modo História

Diversos mundos, compostos por várias fases com características específicas desse mundo. Para desbloquear uma fase, os jogadores devem adquirir, na fase anterior, pontuação suficiente para receberem ao menos 1 das 3 estrelas possíveis por fase. Além disso, algumas fases só podem ser desbloqueadas caso os jogadores possuam uma determinada quantidade de estrelas no total. Caso não possuam o número de estrelas requeridos pela fase, estes jogadores devem retornar às fases anteriores a fim de acrescer a quantia de estrelas adquiridas. Existem também, fases bônus que podem ser utilizadas para coletar estrelas. Após o término da história, é possível jogar novamente as fases da história para tentar adquirir a estrela extra em cada uma dessas fases.

B.6.1.2.2 Modo *Arcade* e Modo Versus

Não há um sistema de progressão, apenas partidas únicas.

B.6.2 **Mecânicas**

B.6.2.1 Física

Dentro das fases, todos os itens possuem física, dessa forma, os chefs colidem com os diversos componentes do cenário como bancadas, panelas, ingredientes, extintores, pratos e entre si.

Durante os diversos níveis, as bancadas da cozinha, as plataformas e escadas podem trocar de posição ou até mesmo desaparecer para aumentar o grau de dificuldade das fases.

B.6.2.2 Ações do Personagem Principal

Cada personagem pode realizar as seguintes ações: Movimentar

- Pegar/Largar itens
- Cortar alimentos

- Arremessar alimentos
- Lavar pratos
- Impulsionar
- Realizar comunicação rápida

B.6.2.3 Combate

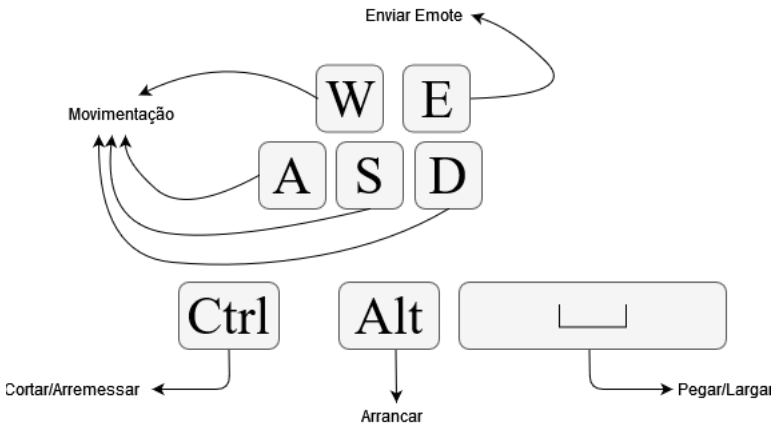
Esse jogo não possui sistema de combate.

B.6.2.4 Sistema de Economia

O sistema de economia está presente dentro de cada fase. Nesse sistema, os jogadores recebem dinheiro à medida que os pedidos são entregues. Caso o tempo para a entrega de um pedido se encerre sem que esse tenha sido preparado e entregue, os jogadores perdem parte da quantia acumulada por eles até o presente momento da fase. Por fim, se os pedidos forem entregues na ordem em que aparecem aos jogadores, há um bônus multiplicador que aumenta o valor recebido por cada pedido. No final da fase, o dinheiro é contabilizado para que se possa avaliar o desempenho dos jogadores por meio de estrelas.

B.6.2.5 *Inputs* Recebidos

Figura 17 – *Inputs*



Fonte: Autores

B.7 Aspectos de Qualidade

B.7.1 Diversão em Grupo

Para alcançar esse objetivo, o jogo deve disponibilizar a possibilidade de se realizar a campanha de maneira cooperativa, onde cada fase necessita de uma boa coordenação

entre os jogadores para que os objetivos sejam alcançados. Outra maneira é a criação de um modo em que os jogadores possam se enfrentar e vencer o outro, promovendo assim a competitividade.

B.7.2 **Desafiador**

Apesar de ter uma temática e ambientação relaxante, o jogo deve fazer com que os jogadores dêem o melhor de si para completar seus diversos níveis. A introdução de novas receitas cada vez mais complexas, padrões variáveis de plataformas, bem como o design das fases são maneiras de alcançar esse objetivo.

B.7.3 **Viciante**

Para prender o foco e a atenção dos jogadores, as fases devem ser curtas para que esses possam jogar as fases continuamente, sem a sensação que estão gastando tempo excessivo com isso. Além disso, a possibilidade de jogar em grupo com amigos, melhora a experiência, fazendo com que o jogador tenda a jogar por mais tempo.

B.8 **História**

B.8.1 **Narrativa**

B.8.1.1 *Background* da História

No reino da cebola, o Rei cebola tenta utilizar os poderes do lendário NECRO-NHAMNHAM-LIVRO e acaba despertando os terríveis pão-demônios. Cabe agora aos seus melhores chefs o trabalho de afugentá-los. Para isso, os chefs botam o pé na estrada para se aprimorarem e aprender novas receitas, a fim de controlar a infestação dos pão demônios.

B.8.1.2 Encerramento da História

Após aprenderem diversas receitas, os chefs servem os famintos pão-demônios, acalmando assim sua fome e trazendo paz para o reino da cebola.

B.8.2 **Personagens**

B.8.2.1 Rei Cebola

B.8.2.1.1 História

O monarca do Reino da cebola, ao retornar de uma de suas aventuras, tomou posse de um misterioso livro e tenta utilizar os poderes contidos nesse livro, apesar dos avisos

de seu cachorro Kevin. Isso liberta as hordas dos pão-demônios.

B.8.2.1.2 Personalidade

Aventureiro e impulsivo.

B.8.2.1.3 Características

É uma cebola, possui bigode, além de ostentar uma coroa e um cajado com uma cebola no topo.

B.8.2.1.4 Papel na Narrativa

Ele liberta a horda dos pão-demônios e agora precisa que seus chefes aprendam novas receitas para impedir os esses monstros.

B.8.2.1.5 Relação com Outros Personagens

Dono do cachorro Kevin, e

É servido pelos chefes de cozinha, os quais ele manda em uma aventura para aprender novas receitas.

B.8.3 ***Cutscenes***

B.8.3.1 *Cutscene 1*

B.8.3.1.1 Personagens

Rei Cebola

Kevin

Chefs de cozinha

Pão-demônios

B.8.3.1.2 Descrição

Após encontrar o NECRO-NHAMNHAM-LIVRO, o Rei cebola tenta usar os segredos contidos no livro.

B.8.3.1.3 *Script*

Imagem da entrada do castelo do Rei Cebola durante uma tempestade.

Narrador: Logo após a hora do chá

Corta para a sala do trono com o Rei Cebola, Kevin e os chefes, com o NECRO-NHAMNHAM-LIVRO à sua frente em um altar.

Rei Cebola: Finalmente! Não acredito no quanto esperei por este momento. As provações pelas quais passamos para receber este prêmio honorável. Olhe só isso, Kevin, olhe só. O lendário... NECRO-NHAMNHAM-LIVRO.

O livro se abre.

Kevin: LATIDO

Rei Cebola: Como é que é?

Kevin: LATIDO

Rei Cebola: Como assim, é perigoso? Não seja tolo! Este livro guarda os segredos da grandiosidade culinária eterna! Em uma tigela grande dissolva o açúcar em água morna, e deixe descansar até que o fermento lembre uma... espuma... cremosa...

Kevin: LATIDO

Rei Cebola: Misture sal e óleo e acrescente a farinha. Amasse a massa... e cubra-a com um pano úmido.

Corta para a imagem de um cemitério com o castelo ao fundo.

Rei Cebola: E deixe... CRESCER!!!

Um raio corta os céus e os pão-demônios começam a se levantar. Corta para à sala do trono novamente e o Rei Cebola está dançando

Rei Cebola: CRESÇA! CRESÇA!!! AHAHAHAHAHAHAHA !!! Huh??

O Rei Cebola olha pela janela

Rei Cebola: Não, não, não! Isso não deveria acontecer! Não pode ser..

Corta para o cemitério

Rei Cebola: OS PÃO-DEMÔNIOS! Como meus melhores chefs, tentem afugentá-los. Eles parecem tão famintos, vamos lá!

Corta para o início da fase

B.9 Níveis e/ou Mundo do Jogo

B.9.1 Níveis

B.9.1.1 Nível 1

B.9.1.1.1 Resumo

A fase se ambienta em uma cozinha com estilo da culinária japonesa e os jogadores devem preparar refeições relacionadas à essa culinária. Os dois pratos disponíveis são:

Peixe cru fatiado ($28 + gorjetas): Os cozinheiros devem pegar o peixe, cortá-lo, colocá-lo em um prato e levar para a entrega, e

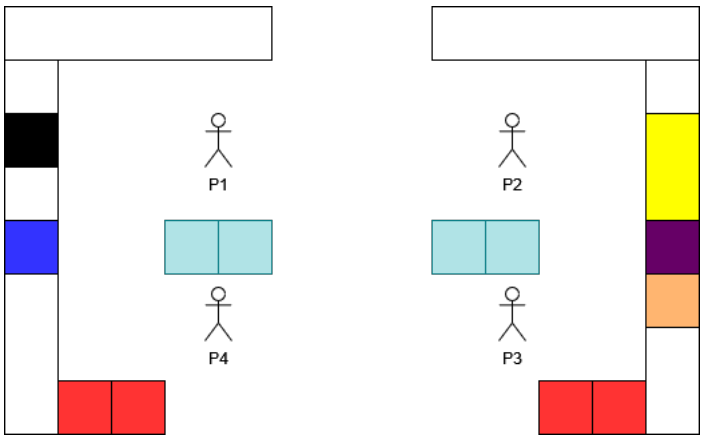
Camarão fatiado ($28 + gorjetas): Os cozinheiros devem pegar o camarão, cortá-lo, colocá-lo em um prato e levar para a entrega.

B.9.1.1.2 Objetivo

Alcançar $20 para 1 estrela, $60 para 2, $240 para 3 e $1300 para a estrela extra.

B.9.1.1.3 Mapa

Figura 18 – *Mapa da Fase 1-1 - Overcooked 2*



Fonte: Autores

Branco: bancadas em que os itens podem ser depositados;

Preto: lixeira onde itens podem ser descartados;

Azul escuro: caixa de onde são retirados os peixes para preparo das refeições;

Vermelho: bancadas de corte onde os ingredientes podem ser fatiados;

Laranja: caixa de onde são retirados os camarões para preparo das refeições;

Roxo: local onde os pratos reaparecem após a entrega;

Amarelo: local de entrega das refeições após a finalização do preparo, e

Azul claro: bancadas em que os pratos estão posicionados ao início da fase.

B.9.1.1.4 Conclusão

Finalizado o tempo da fase, o dinheiro adquirido é contabilizado e superando o mínimo necessário, o jogador recebe 1, 2 ou 3 estrelas de acordo com o total obtido. Caso contrário, a fase deverá ser jogada novamente.

B.9.2 **Mundo do jogo**

B.9.2.1 Visão Geral do Mundo

O mundo do jogo é composto por 6 áreas que se conectam ao castelo, não havendo nenhum tipo de conexão entre si. Dentro dessas áreas há diversos tipos de ambientes como ar, água, mina, entre outros. Os jogadores podem se movimentar livremente pelas áreas, porém as fases são desbloqueadas conforme o progresso do jogador(es).

B.9.2.2 Área 1

B.9.2.2.1 Descrição Geral

Contém as primeiras fases do jogo, com apresentação das principais mecânicas.

B.9.2.2.2 Características Específicas

Combina níveis de dois temas distintos, sendo a culinária japonesa e missões aéreas tripulando um balão. São fases com o nível de dificuldade mais baixo, tendo em vista que são introdutórias para os jogadores.

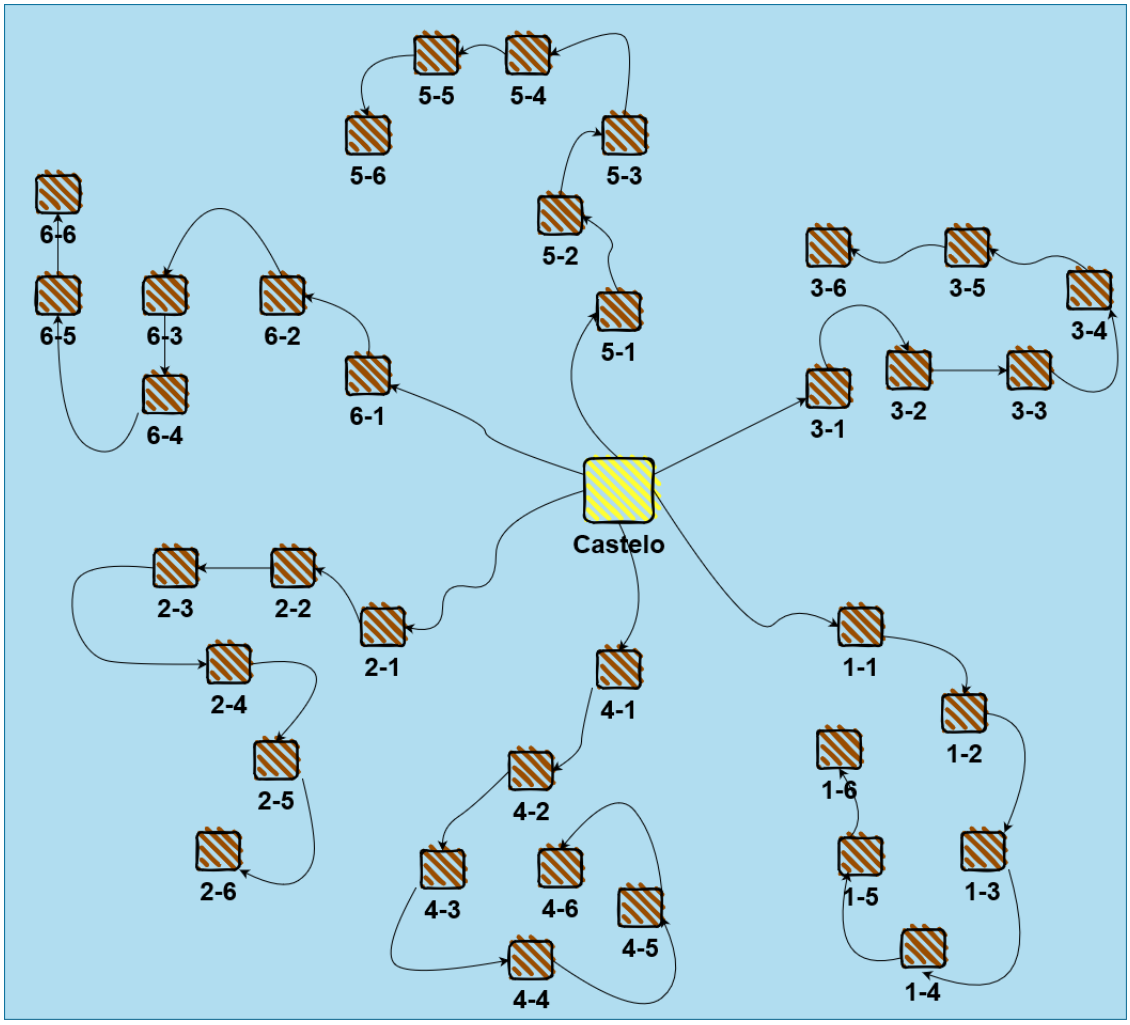
B.9.2.2.3 Conexão com Outras Áreas

Essa área se conecta somente com o castelo.

B.9.2.2.4 Níveis (se houver) que utilizam essa área

Os níveis 1-1, 1-2, 1-3, 1-4, 1-5 e 1-6 compõem essa área.

Figura 19 – *Mapa - Overcooked 2*



Fonte: Autores

B.10 Interface

B.10.1 Menus

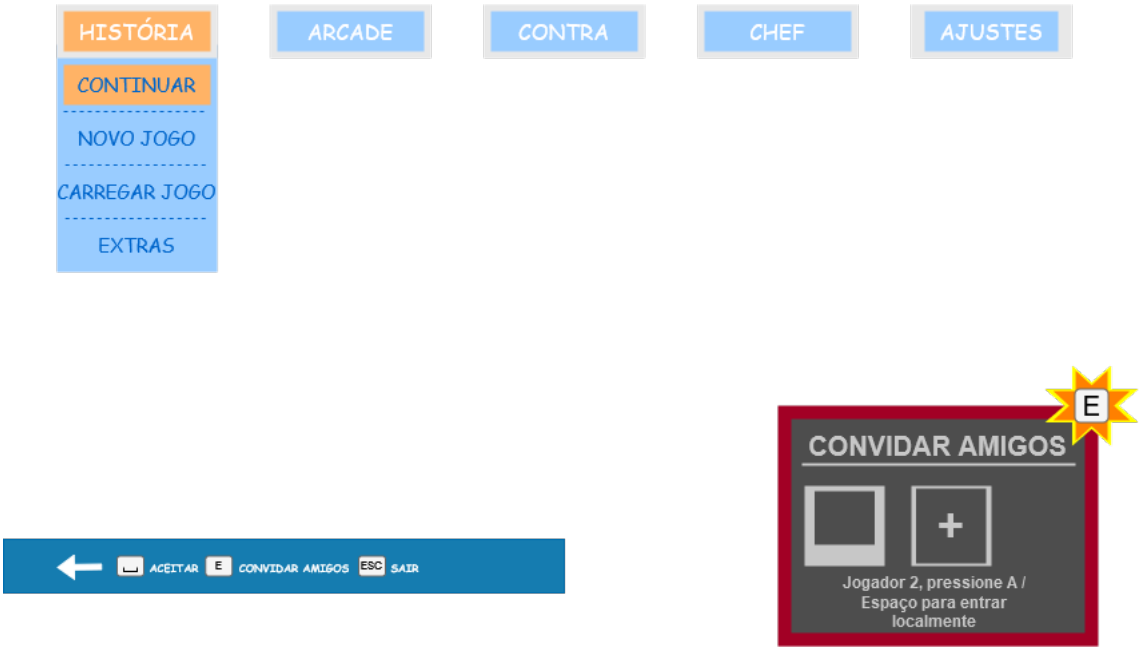
B.10.1.1 Principal

O menu principal consiste de três grandes áreas. Na área superior estão as principais opções do jogo, sendo história, arcade e contra, que são modos de jogo, e chef e ajustes que se tratam de configurações do jogo. No canto inferior direito está a zona para convidar amigos e na esquerda é mostrado alguns comandos úteis e suas respectivas teclas.

B.10.1.2 Pausa

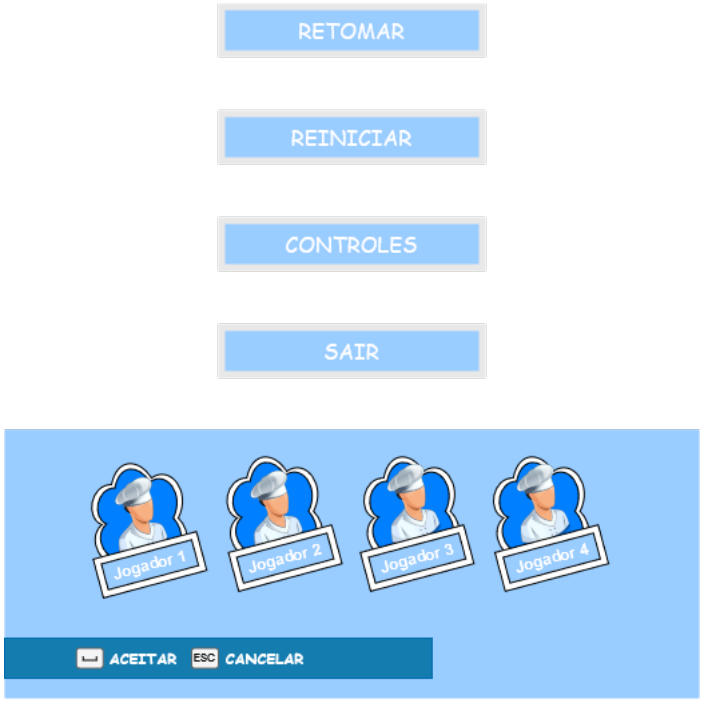
O menu de pausa possui duas variações: quando o jogador está no mapa principal (Primeira imagem) ou quando ele está dentro de uma fase (Segunda imagem). Três opções (Retomar, Sair e Controles) aparecem em ambos os menus, sendo a principal diferença a presença de uma opção para reiniciar a fase quando o jogador está jogando um nível,

Figura 20 – *HUD - Menu Principal*



Fonte: Autores

Figura 21 – *HUD - Menu de Pausa*



Fonte: Autores

enquanto a opção de mudar de chef aparece quando o jogador se encontra no menu principal.

B.10.2 *Head-up displays (HUDs)*

B.10.2.1 Mapa

Figura 22 – *HUD - Mapa*

Fonte: Autores

O HUD do mapa apresenta duas seções, ambas localizadas na parte superior da tela. Na esquerda, uma bússola para orientar os jogadores que podem se movimentar livremente no mapa e na direita o modo de jogo (história, treino e sobrevivência) e a quantidade de estrelas obtidas.

B.10.2.2 Fases

Figura 23 – *HUD - Fase*

Fonte: Autores

O HUD das fases é composto por três ambientes. A seção localizada na área superior da tela apresenta os pedidos em ordem de chegada, em cada um, estão registrados os ingredientes que compõem aquele prato e o modo de preparo. No canto inferior esquerdo é mostrado o valor em pedidos, bem como a barra do multiplicador de gorjetas obtidas. Por fim, no canto inferior direito é ilustrado o tempo ainda disponível para realização do nível.

B.10.3 **Sistemas de ajuda (tutoriais)**

Durante a fase inicial do jogo (Fase Tutorial), as explicações são apresentadas em uma caixa de texto na parte superior central da tela, enquanto aparecem marcações nos

locais onde os jogadores devem realizar as ações, como cortar os ingredientes na bancada de corte, inserir os ingredientes cortados nos pratos e realizar a entrega da refeição pronta.

B.11 Arte e Vídeo

B.11.1 Conceito de arte

Seguindo o modelo de arte do primeiro jogo da série (Overcooked), conta com um estilo simples, no qual não há um grande detalhamento das texturas presentes durante a *gameplay*. A ideia é trazer um estilo mais agradável para os diferentes públicos que o jogo busca atingir, desde os mais novos até os mais velhos, alcançando assim o conceito de um jogo "mais família". As diversas fases do modo história buscam acompanhar o esquema de arte de cada área presente no mapa principal do jogo. Por exemplo, as fases aéreas buscam seguir essa temática, ao mesmo tempo que é feita uma transição agradável para a próxima área.

B.11.2 2D

Menu de pausa

Telas de carregamento

Logo do jogo

Elementos da interface (pedidos, gorjeta obtida, tempo)

B.11.3 3D

Menu principal

Mapa principal do modo história

Ambientes de cada nível

Ingredientes

Personagens

Bancadas de cozinha

Utensílios de cozinha

B.11.4 Cinemáticas

Cutscenes

B.12 Música e Efeitos Sonoros

B.12.1 Proposta do *Design* de Som

A principal proposta do design de som do jogo é trazer um ambiente tranquilo e acolhedor para os jogadores. Tem o objetivo de relaxar os usuários para que esses possam ter uma experiência o menos estressante possível.

Tanto as músicas quanto os efeitos sonoros são resultados do trabalho de Oliver Wood e Danny Hey.

B.12.2 Músicas

As músicas utilizadas nos menus e na introdução do jogo buscam atingir um aspecto mais suave e relaxante. De forma a acolher e tranquilizar os jogadores, essas músicas incorporam sons da natureza, como o canto dos pássaros.

Ao se iniciar cada fase, é reproduzida uma música ambiente, semelhante ao jazz para que os cozinheiros possam realizar suas atividades na cozinha com muita tranquilidade. Porém, ao chegar nos instantes finais de cada fase, a música ambiente se acelera, a fim de instigar os jogadores a entregarem a maior quantidade possível de pratos antes do término da fase.

Por fim, há uma música ao final de cada nível, para fazer a transição de um ambiente mais agitado do final de uma fase, para um ambiente mais tranquilo, como um menu ou o mapa principal da história.

B.12.3 Efeitos Sonoros

Sons quando o ônibus se locomove no mapa (ex: passar na água, abrir asas em ambientes que se passam no ar, entre outros)

Sons quando o usuário interage com o menu (ex: passar o cursor em algumas da opções, selecionar uma opção)

Durante os níveis, efeitos sonoros para as tarefas da cozinha (ex: cortar alimentos, lavar louças)

Sons de aviso que um dos pratos está para queimar

Sons punitivos (ex: quando os jogadores entregam pedidos fora de ordem)

Som de relógio quando o tempo do nível estiver acabando

B.13 Sistemas Alvo

Abaixo são apresentadas as plataformas em que o jogo poderá ser jogado, bem como os requisitos de *hardware* mínimos necessários em cada plataforma.

B.13.1 Windows

Sistema Operacional: WIN7-64 bit

Processador: Intel i3-2100 / AMD A8-5600k

Memória: 4 GB de RAM

Placa de vídeo: GeForce GTX 630 / Radeon HD 6570

DirectX: Versão 11

Armazenamento: 3 GB de espaço disponível

B.13.2 MacOs

Sistema Operacional: MacOS Sierra - 10.12.6

Processador: 2.7 GHz Intel Core i5

Memória: 4 GB de RAM

Placa de vídeo: Nvidia GeForce GT 640M

Armazenamento: 3 GB de espaço disponível

B.13.3 Linux

Sistema Operacional: Mint 18 / Ubuntu 16.04.01

Processador: Intel i3-2100 / AMD A8-5600K

Memória: 4 GB de RAM

Placa de vídeo: GeForce 450GTS

Armazenamento: 3 GB de espaço disponível

Placa de som: DirectX Compatible Sound Card

B.13.4 PlayStation 4/ PlayStation 5/ Xbox One/ Xbox Series X/ Nintendo Switch

Configuração de *hardware* padrão de cada plataforma.

B.14 Cronograma

B.14.1 Data de Início do Projeto

xx/xx/xxxx

B.14.2 Ambiente 1

B.14.2.1 Descrição

Criação dos seis níveis pertencentes ao ambiente 1 do mapa.

B.14.2.2 Objetivo

Concluir os seis níveis com todas as mecânicas envolvidas e a arte completa.

B.14.2.3 Prazo

5 meses.

B.14.3 Data de Conclusão de Projeto

xx/xx/xxxx

B.14.4 Priorização - MoSCoW

Figura 24 – *MoSCoW*

ID	Descrição	Comentários	Must	Should	Could	Would
1	Criar menu		X			
2	Ambiente 1		X			
3	Ambiente 2		X			
4	Fases bônus			X		
5	Atualizações sazonais	DLC será disponibilizada após o lançamento do jogo base			X	

Fonte: Autores

B.14.5 **Referências**

Aqui devem ser colocadas as referências utilizadas de outros jogos. São sugeridas duas maneiras:

Na primeira, coloque o nome do jogo referência e ao que se refere no seu jogo. Exemplo:

Cooking simulator, Big Cheese Studio , 2019. Comparativo;

Na segunda, numere suas referências e coloque essa numeração no trecho do documento ao qual ela se refere. Exemplo:

[1] Cooking simulator, Big Cheese Studio , 2019.

[2] Jogo Y, 1997.

[3] Jogo Z, 2017.