



Universidade Federal de Pernambuco

Centro de Informática

Pós Graduação em Ciência da Computação

Game Live Logs: Uma Plataforma de Conversação para Atenuar
Conflitos no Desenvolvimento de Games

Aluno: Tiago Lemos de Araujo Machado (tlam@cin.ufpe.br)

Orientador: Prof. Geber Lisboa Ramalho (glr@cin.ufpe.br)

Co-orientadora: Prof^a Carina Frota Alves (cfa@cin.ufpe.br)

Recife, Setembro de 2013

Agradecimentos

Primeiramente, aos meus pais, Julia & Adeilton e aos meus irmãos Alysson & Diana.

Aos amigos de Gravatá pelas incontáveis horas de diversão.

À professora Cristiane Conde pelos inúmeros elogios às minhas redações.

À direção do Colégio Vitória e as amigas e amigos da cidade de Vitória de Santo Antão.

Aos amigos da Graduação no Centro de Informática e da UFPE como um todo, pelos ensinamentos e momentos de descontração.

Aos funcionários do Centro de Informática que em todos os meus dias de estudo e trabalho me deram o suporte necessário a todas as minhas atividades.

À professora Renata Souza pela oportunidade de monitoria no departamento de Estatística.

Ao professor Marcelo Walter pela colaboração na minha Iniciação Científica.

Ao professor Alex Sandro Gomes pelas inúmeras oportunidades nas quais trabalhamos juntos nos últimos anos.

À Aline Timóteo pela chance de trabalhar no projeto Lampejo.

Ao professor Geber Ramalho pelos vários papers que publicamos, pela orientação no meu Trabalho de Graduação, por ter aceitado continuar como meu orientador no Mestrado e sobretudo, por ter compartilhado suas questões de pesquisa comigo, me deixando muito honrado e agradecido.

À Renan Lima e Fábio Florêncio pelo paper do SBGAMES 2012.

À Anderson Paulo, Luiz Francisco, Felipe Borgiani, Gutenberg Barros, Vinicius Fabrino, Danielle Gomes, Eduardo Oliveira, Pedro Alessio e Nilson Soares pela excelente equipe de trabalho no CESAR/NAVE.

Aos alunos do Cícero Dias (NAVE Recife) pelo exemplo e inspiração.

À Túlio da Manifesto, Márcio da D'accord e todos os colaboradores da pesquisa.

À Professora Carina por aceitar co-orientar meu Mestrado.

Aos professores André Neves e Vinicius Garcia por terem aceitado participar da minha avaliação e pelas valiosas recomendações para o futuro deste trabalho.

Resumo

O processo de desenvolvimento de Games é um trabalho difícil [Blow 2004]. Envolve muitas disciplinas e contém inúmeras fases e tarefas. Nas duas últimas décadas, o crescimento da indústria tem sido evidente, o que motivou uma série de pesquisas acerca de como a indústria encara tal processo. Alguns desses estudos relatam problemas nos processos de produção que podem ser uma barreira a impedir o futuro crescimento da indústria [Potanim 2010]. Notamos que muitos desses problemas estão associados a conflitos entre as fases de pré-produção e produção, para as quais a literatura diverge em vários pontos e não oferece respostas adequadas. Um dos conflitos observados diz respeito aos artefatos de suporte a produção. Não há uma padronização sobre quais são necessários, quais seus conteúdos e como devem ser apresentados. Tradicionalmente, de acordo com a literatura, o Documento de Game Design (GDD) teria a função de ser o principal artefato, o guia para equipe de produção criar o software do Game propriamente dito, quase como um roteiro cinematográfico ou a planta baixa de um projeto de arquitetura [Schuytema 2006]. Entretanto, de acordo com nossas pesquisas envolvendo membros da indústria de games (nacional e internacional), a importância de tal artefato foi questionada. Considerando tais pontos, o objetivo deste trabalho foi analisar os conflitos que cercam a pré-produção e a produção de games, de acordo com o ponto de vista da indústria por meio de questionários e entrevistas, contando com participantes atuantes no Brasil e no exterior. A partir do discurso dos entrevistados, definimos um conjunto de princípios, que posteriormente deram origem a uma proposta para atenuar os problemas expostos, tal proposta se traduziu em uma ferramenta, que foi avaliada por uma equipe de profissionais durante o desenvolvimento de um game. Ao longo de quatro semanas, os mesmos ofereceram críticas e sugestões que permitiram a evolução da ferramenta, intitulada Game Live Logs.

Palavras-chave: Documento de Game Design, Pré-produção, Produção de Games

Abstract

The game development process is a difficult work [Blow 2004]. Involves many disciplines and contains numerous phases and tasks. In the last two decades, the industry's growth has been evident, which motivated a lot of research on how the industry sees this process. Some of these studies reported problems in production processes that can be a barrier to prevent the future growth of the industry [Potanim 2010]. We note that many of these problems are associated with conflicts between the stages of pre-production and production, for which the literature differs on several points and does not provide adequate answers. One of the conflicts observed with respect to supporting artifacts production. There is no standardization about what is needed, what their content and how they should be presented. Traditionally, according to the literature, the Game Design Document (GDD) would be the main artifact, the guide for production team to create the game software itself, almost like a movie script or an architecture blueprint design [Schuytema 2006]. However, according to our research involving members of the game industry (national and international), the importance of such an artifact was questioned. Considering these points, the objective of this study was to analyze the conflicts surrounding the pre-production and production of games, according to the view of the industry through questionnaires and interviews, with participants working in Brazil and abroad. From the interviewees' discourse, we define a set of principles, which subsequently led to a proposal to mitigate the above problems, this proposal has resulted in a tool that was evaluated by a team of professionals during the development of a game. Over four weeks, they offered criticisms and suggestions that allowed the development of the tool, entitled Game Live Logs.

Key-words: Game Design Document, Pre-Production, Game production

Sumário

1. Introdução	14
1.1 Motivação	15
1.2 Objetivos	15
1.3 Escopo negativo	15
1.4 Resultados esperados	15
1.5 Estrutura do trabalho.....	16
2. A documentação de games e sua influência na produção	17
2.1 Definição e papel do GDD.....	17
2.2 Formato tradicional do GDD	20
2.3 Criticas ao GDD.....	22
2.3.1 Os Problemas da documentação identificados em estudos de documentos de <i>Game Postmortens</i>	23
2.3.2 Problemas encontrados nos processos de desenvolvimento diretamente decorrentes da documentação.....	25
2.3.3 Problemas na documentação e sua influência nos custos dos projetos	27
2.4 Nossas pesquisas preliminares: primeiro estudo junto à indústria	28
2.5 Diagnóstico: compilação de Inadequações e Conflitos do GDD.....	31
2.3.1 Público – Alvo diversificado.....	31
2.3.2 Diversidade de formas e conteúdos.....	31
2.3.3 Imposição do GDD por culturas corporativas	31
2.3.4 Evolução do GDD	32
2.3.5 O GDD e o Plano de Projeto	32
3 O problema do ponto de vista da indústria: entrevistas.....	34
3.1 Participantes do Estudo.....	35
3.2 Entrevistas Semi Estruturadas	37

3.3 Procedimento de Análise: Análise Qualitativa de Conteúdo	38
3.4 Resultados das Entrevistas	40
3.4.1 Público – Alvo do GDD	40
3.4.2 Forma e Conteúdo do GDD	41
3.4.3 Cultura Corporativa.....	42
3.4.4 Evolução do GDD	42
3.4.5 O GDD e o Plano de Projeto	43
3.4.6 Game como Documento	44
3.5 Conclusões sobre os resultados das entrevistas	45
3.5.1 Investimento em pré-produção	45
3.5.2 Gerenciamento das características do game.....	45
3.5.3 Código versus documento	46
3.5.4 Academia versus indústria	46
4 Princípios para uma nova documentação de games	48
4.1 Novo diagnóstico: atualizando a compilação de inadequações e conflitos do GDD.....	48
4.1.1 Público – Alvo diversificado	48
4.1.2 Diversidade de formas e conteúdos	49
4.1.3 Imposição do GDD por culturas corporativas	49
4.1.4 Evolução do GDD	49
4.1.5 A Documentação e o Plano de Projeto	49
4.1.6 O GDD como ferramenta para gestão de conhecimento	49
4.1.7 Código como documento	50
4.2 Inspirações	50
4.3 Requisitos	52
4.4 Classificação e análise dos Requisitos	56
4.4.1 Requisitos Funcionais	56

4.4.2 Requisitos Não Funcionais.....	58
5 Game Live Logs	61
5.1 Influência	61
5.2 Running Lean e Metodologias Ágeis.....	61
5.3 Protótipo versão 0 (V.0).....	62
5.4 Primeira Avaliação	63
5.5 Primeiros Resultados	63
5.6 Framework DECIDE	66
5.7 Ciclo de aprendizado	67
5.7.1 Sujeitos	67
5.7.2 Procedimento das avaliações.....	67
5.7.2.1 Etapa 1 – Conhecimento e contextualização.....	67
5.7.2.2 Etapa 2 – Testes propriamente ditos	68
5.7.2.3 Protocolo Cíclico de Avaliações	68
5.8 Evolução do Protótipo	69
5.8.1 Primeira visita	71
5.8.2 Evolução do Protótipo – Primeira Semana	72
5.8.3 Segunda visita	75
5.8.4 Evolução do Protótipo – Segunda Semana	75
5.8.5 Terceira visita.....	77
5.8.6 Evolução do Protótipo – Terceira semana.....	77
5.8.7 Quarta visita	79
5.8.8 Evolução do Protótipo – Quarta semana.....	80
5.9 Comparativo de ferramentas	81
5.10 Avaliação dos usuários afoitos	83
5.10.1 Facilidade no Uso.....	84
5.10.2 Melhorias sugeridas.....	84

5.10.3 Cenários.....	85
5.11 Conclusões da Avaliação	85
5.11.1 Público – Alvo diversificado.....	86
5.11.2 Diversidade de formas e conteúdos.....	87
5.11.3 Imposição do GDD por culturas corporativas.....	87
5.11.4 Evolução do GDD	87
5.11.5 Documentação e o plano de projeto	88
5.11.6 Código como documento	88
6 Implementação: Definições técnicas	90
6.1 Game Design nas nuvens.....	90
6.2 Google App Engine e a estrutura do projeto.....	90
6.2.1 Backend.....	91
6.2.1.1 Camada Models.....	91
6.2.1.2 Camada Managers	92
6.2.1.3 Camada Controls	93
6.2.2 Frontend	93
6.2.3 Comunicação Backend – Frontend	94
6.3 Visualização e representação da interação do usuário com o sistema	95
6.3.1 Pacote Models	95
6.3.2 Pacote Controls	96
6.3.3 Pacote Managers	96
6.4 Operação de inserir um Log.....	97
7 Conclusões.....	100
7.1 Contribuições	100
7.1.1 Crítica ao GDD.....	100
7.1.2 Criação do Game Live Logs.....	101

7.1.3 Valorização da Análise e Observação do Problema para o Contexto Computacional	101
7.2 Trabalhos Futuros	102
Referências	104

Lista de Figuras

Figura 2.1: Documento do game <i>Shooter: Majestic Revelations</i>	20
Figura 2.2: Fragmento do capítulo sobre mecânicas do documento de uma das versões do game <i>Wing Commander</i>	21
Figura 2.3: Fragmento do documento do game <i>Halo</i>	21
Figura 2.4: Fragmento do documento do game <i>Bayonetta</i>	22
Figura 3.1: Interface do Atlas.ti	40
Figura 5.1: O formato do Game Design Logs [Cook, D 2011].	63
Figura 5.2: Visualização de um log no nosso protótipo versão zero.	64
Figura 5.3: A lista de logs e seus anexos associados..	65
Figura 5.4: protótipo da interface para inserir	66
Figura 5.5: protótipo da visualização dos logs.	66
Figuras 5.6 e 5.7: interface para inserir um <i>log</i> e interface de visualização de um <i>log</i> . ..	71
Figura 5.8: as formas de consulta disponíveis	72
Figura 5.9: usuário explica através de um esboço como gostaria de editar o texto dos logs.	72
Figuras 5.10 e 5.11	75
Figura 5.12: tags pré definidas	78
Figura 5.13: Formulário de criação de Logs, agora modificado com o campo para indicar tarefa associada em ferramenta de gerenciamento de projetos.	81
Figura 5.14: Opção para formatar texto como código	82
Figura 5.15: Painel de controle da App Engine.	87
Figura 6.1: Classes do pacote <i>models</i>	96
Figura 6.2: Classes do pacote <i>controls</i>	97
Figura 6.3: Classes do pacote <i>mangers</i>	97
Figura 6.4: formulário para inserir um <i>log</i>	95
Figura 6.5: fluxo das informações no sistema.	100

Lista de Tabelas

Tabela 2.1: Os elementos mais importantes do GDD	18
Tabela 4.1: Dados da pesquisa de [Souza 2009] indicando as vantagens e desvantagens no uso de três formatos para elaboração de GDDs.....	52
Tabela 4.2: Mapeamento de princípios e requisitos para o diagnóstico do público alvo.	54
Tabela 4.3: Mapa de princípios e requisitos para o diagnóstico de formas e conteúdo.	55
Tabela 4.4: Mapa de princípios e requisitos para o diagnóstico de culturas corporativas.	55
Tabela 4.5: Mapeamento de princípios e requisitos para o diagnóstico da evolução do GDD. ...	56
Tabela 4.6: Mapeamento de princípios e requisitos para o diagnóstico do diálogo do GDD com o plano de projetos.	56
Tabela 4.7: Mapeamento de princípios e requisitos para o diagnóstico código como documento.	57
Tabela 5.1: problemas encontrados e melhorias sugeridas na primeira semana	73
Tabela 5.2: problemas encontrados e melhorias sugeridas na segunda semana.....	76
Tabela 5.3: problemas encontrados e melhorias sugeridas na terceira semana.....	78
Tabela 5.4: problemas encontrados e melhorias sugeridas na quarta semana.....	81
Tabela 5.5: As ferramentas e impressões sobre as características apresentadas	83

Lista de Quadros

Quadro 3.1: Resumo dos objetivos da pesquisa	38
Quadro 5.1: Características do Protótipo V.1	70
Quadro 5.2: Características do Protótipo V.2	73
Quadro 5.3: Características do Protótipo V.3	77
Quadro 5.4: Características do Protótipo V.5	79

Lista de Gráficos

Gráfico 2.1: Lista de problemas e número de ocorrências dos mesmos.....	24
Gráfico 2.2: importância do Game Designer no processo de implementação.....	28
Gráfico 2.3: Níveis de concordância em relação a geração de artefatos complementares ao GDD	29
Gráfico 3.1: Papéis dos participantes da pesquisa	36
Gráfico 3.2: Papéis dos participantes da pesquisa (considerando Produtores que atuam como Game Designers).....	37

1. Introdução

Embora sua presença nos ciclos de desenvolvimento seja considerada como um dos elementos mais importantes para a produção do game, temos observado que a importância do documento de projeto conceitual do jogo (GDD - do inglês Game Design Document) tem se revelado cada vez mais contraditória. Considerado pela literatura [Schuytema 2006; Schell 2008] como um dos principais elementos no processo de produção de games, percebemos ao longo das nossas pesquisas que na prática tal documento está no coração de uma série de conflitos entre pré-produção e produção na indústria. Alguns produtores e designers [Richard Rouse III 2004], questionam a efetividade do documento, pois segundo seus trabalhos, a tarefa de expressar a quantidade de elementos de um game e suas interações através de tal artefato não é prontamente atendida por muitos Game Designers. Inicialmente, como indica o capítulo de sua obra dedicado a documentação [Richard Rouse III 2004], o desenvolvimento de games possuía times compostos por apenas um indivíduo multitarefa. Por isso a função de documentar (pelo menos em termos formais) o projeto não era de grande importância, pois se a pessoa conseguia estabelecer e implementar sozinha sua visão do game (considerando as limitações de software e hardware da época), pouco importava se isso estava (bem) documentado ou não. O próprio game já era, em si, um documento.

Atualmente a necessidade de documentação nos projetos de games é um efeito do crescimento da indústria, que alcançou sobretudo nas últimas décadas uma escala comparável a das grandes produções cinematográficas [Hirsch et al. 2007]. O indivíduo multitarefa de Richard Rouse III deu lugar à equipes multidisciplinares [Barros 2009] que envolve especialistas em marketing, engenheiros, designers, produtores, artistas, *testers*, entre tantos outros. Atualmente, devido a tantas especialidades e tarefas diversas, manter a visão do game coerente entre todos os membros da equipe e entre todas as fases da produção passou a ser um trabalho necessário e de grande complexidade. Temos visto em pesquisas [Callele et al. 2005; Winget et al. 2011] que há muitas divergências quanto ao conteúdo, forma e tamanho do artefato gerando dilemas como o mencionado por um profissional abaixo.

“Se eu escrever um documento longo, ninguém o lerá porque será muito cansativo e se eu escrever um documento curto, todos terão um monte de dúvidas e irão me questionar

o tempo todo sobre coisas que eu ‘deveria’ ter escrito ou que necessitam de melhor explicação.”

1.1 Motivação

A motivação deste trabalho diz respeito a compreender, junto a indústria, quais os principais problemas referentes ao GDD que prejudicam seus processos e se propagam nas etapas do projeto, sobretudo no intervalo entre a pré-produção e a produção.

1.2 Objetivos

Os objetivos deste trabalho consistem em determinar um diagnóstico acerca dos problemas decorrentes do GDD bem como suas propagações entre as etapas de produção, e utilizar o conhecimento obtido com a pesquisa para propor uma ferramenta de apoio que consiga atenuar tais conflitos. Para isso, realizamos uma revisão da literatura, reutilizamos resultados de nossa pesquisa anterior [Machado 2010], conduzimos uma série de entrevistas com profissionais da indústria, definimos um conjunto de princípios com base nas suas respostas, que por sua vez, serviram de requisitos para construção de um protótipo de ferramenta de documentação, que foi avaliado e evoluído por uma equipe de desenvolvimento de games.

1.3 Escopo negativo

Não discutiremos acerca das práticas usadas por desenvolvedores independentes, a não ser em casos relatados como influências para técnicas usadas na indústria e estritamente mencionadas pelos participantes da pesquisa, também não é nossa intenção discutir as diferenças quanto ao porte das produções nos casos citados. Nos interessam apenas as semelhanças nas dificuldades encontradas tanto por produções de games casuais como produções de games AAA, que são igualmente afetadas pelos problemas do nosso objetivo.

1.4 Resultados esperados

Esperamos prover um diagnóstico apurado acerca do nosso estudo junto a indústria sobre os problemas enfrentados na documentação de games, além de ter uma

ferramenta que transforme esse conhecimento em meios para se evitar tais problemas em produções futuras.

1.5 Estrutura do trabalho

O trabalho está dividido da seguinte forma, o capítulo **2** trata do que é o GDD, seus problemas e uma pesquisa preliminar sobre o tema, o capítulo **3** trata de uma nova pesquisas acerca de como a indústria encara tais problemas, o capítulo **4** traz um conjunto de princípios baseados nos resultados da entrevista, o capítulo **5** detalha nossa contribuição, o Game Live Logs (uma ferramenta para atenuar os conflitos provenientes do GDD). No capítulo **6**, revelamos os detalhes técnicos da implementação do GLL. Por fim o capítulo **7** traz nossas contribuições e trata de como avançar com este projeto em trabalhos futuros.

2. A documentação de games e sua influência na produção

Nos processos de desenvolvimento de Jogos não há nenhum documento que seja mais importante que o GDD. Seu conteúdo tem a pretensão de apresentar todas as informações necessárias para o desenvolvimento do game, chegando em muitos casos a atingir um número bastante expressivo de páginas [Carvalho, 2006]. Isso se deve a enorme quantidade de detalhes e disciplinas envolvidas na produção: Design, Artes, Tecnologia, Gerenciamento de Projetos, Psicologia, Marketing entre tantas outras. Há um esforço enorme por parte dos Designers em organizar todas essas disciplinas de forma a criar uma experiência interativa [Scuytema, 2006] que possa atingir uma diversidade de propósitos, como educação, treinamento, saúde, publicidade e principalmente: emoção e diversão. Obviamente, devido a essa variedade de propósitos e por ser uma área multidisciplinar existem várias maneiras de se escrever um Documento de Game Design [Schell, 2008]. Vários fatores definem como será o Documento de Game Design como o estilo de jogo, a data estabelecida para o lançamento, as políticas da empresa nas quais o jogo será desenvolvido, etc. Todos esses itens exercem certo nível de influência na escrita do documento e seus elementos, que podem ser escritos todos em uma única versão ou então em várias versões do documento com atualizações periódicas [Scuytema, 2006].

2.1 Definição e papel do GDD

Servir como meio de conceituar, descrever e comunicar o registro de ideias acerca do que deve ser o game a ser produzido é a definição típica para o GDD [Rouse III 2001]. Observando o trabalho de outros profissionais [Bates & Robert 2004; Schuytema 2006; Schell 2008; Perry 2009], vemos que tal definição pouco tem variado e a presença de tal documento dentro dos processos de produção, pelo menos na visão da literatura, é inquestionável.

Toda a conceituação, descrição e comunicação é distribuída no GDD por meio de elementos que tratam de objetos específicos do game, como: personagens, interface, controles e etc. A seguir apresentamos alguns desses elementos e suas definições de acordo com estudo realizado tendo base na literatura [Scuytema 2006; Rollings 2003; Schell 2008], no processo de desenvolvimento de empresas de games do Porto Digital [Porto Digital 2009], de empresas internacionais e de Designers e Desenvolvedores do Centro de Artes e Comunicação e do Centro de Informática da UFPE [UFPE 2009],

respectivamente. Profissionais dos locais citados votaram nos elementos que consideraram como os mais importantes para a produção do GDD e do game em si [Tabela 2.1].

Tabela 2.1: Os elementos mais importantes do GDD

Elemento do GDD	Votos (em %)
Visão Geral (High Concept) do game	97
Regras do game	88
Controles (ações do jogador com o personagem)	85
Fluxo do game	82
Estilo (ação, estratégia, puzzle)	73
Formas de pontuação	61
Personagens (controláveis ou não pelo jogador)	55
Referências (filmes, jogos, livros, etc.)	48
Telas do game	30
Linha de arte	30
Itens	30
Design de níveis	30
Tema (idéia filosófica)	27
Análise de competidores	21
Efeitos sonoros	6
Trilha sonora	3

- ***Visão geral (High Concept) do Jogo***

A idéia inicial do Jogo, que apresenta a todos os envolvidos na produção do que se trata o projeto, seus propósitos e objetivos a serem alcançados.

- ***Regras do Jogo***

Todas as regras do jogo, se possível, com suas definições matemáticas. Têm a função de explicar quem ganha, quando ganha, porque ganha, etc.

- ***Controles (ações do jogador com o personagem)***

Todos os comandos disponíveis para executar as ações do personagem controlável pelo jogador.

- ***Fluxo do game***

Descrição individual do gameplay com o máximo de informação possível sobre a colocação dos enigmas, itens e inimigos do jogo de forma a balancear a experiência do jogador.

- ***Estilo (ação, estratégia, puzzle)***

Define qual será o estilo (gênero) do jogo. Corrida, Tiro, Ação ou até mesmo uma junção de vários dos estilos disponíveis.

- ***Formas de pontuação***

Explica todas as formas possíveis de se pontuar no Jogo. A pontuação é sequencial? O Jogo terá alguma forma de atribuir pontos bônus aos jogadores? O jogador poderá perder seus pontos conquistados?

- ***Personagens (controláveis ou não pelo jogador)***

Descrição dos personagens jogáveis e dos não jogáveis (Non Player Character – NPC)

- ***Referências (filmes, jogos, livros, etc.)***

Apresentar referências que auxiliem a compreensão do que é pretendido pelo Design do game em qualquer das disciplinas do desenvolvimento de game.

- ***Telas de Jogo***

As telas de início, de seleção de personagens, Heads Up Display – HUD, fim de jogo (game over), tutoriais, etc.

- ***Linha de arte***

O estilo de arte que definirá o visual do game.

- ***Itens***

Os objetos que podem beneficiar/prejudicar o jogador ao serem utilizados em determinados momentos do game.

- ***Design de níveis***

As descrições sobre o conteúdo dos diversos níveis que o jogador atravessará no decorrer do Jogo. Quem são os inimigos/amigos que ele encontrará? Quais os itens que ele poderá utilizar em tal nível? Há um chefe no final do nível?

- ***Tema (ideia filosófica)***

Associado diretamente ao estilo do jogo. Exemplo: caso o estilo do Jogo seja corrida, o tema pode ser Formula 1. Ou se o estilo do Jogo for um First Person Shooter (FPS), seu tema pode ser II Guerra Mundial.

- **Análise de competidores**

Analisa jogos similares ao que está em desenvolvimento para identificar oportunidades de melhoras e ganhos no processo e na qualidade final do game.

- **Efeitos sonoros**

Os efeitos correspondentes sonoros a ação direta/indireta do game.

- **Trilha sonora**

As músicas que definem a sonoridade e sentimento do game.

2.2 Formato tradicional do GDD

Tradicionalmente, essas informações são disponibilizadas em longos documentos de texto (ver Figuras 2.1, 2.2 e 2.3) escritos durante a fase de pré-produção e fornecidos ao time de desenvolvimento como um guia para orientar a produção [Winget & Sampson 2011; Alves et al. 2006; Callele et al. 2005].

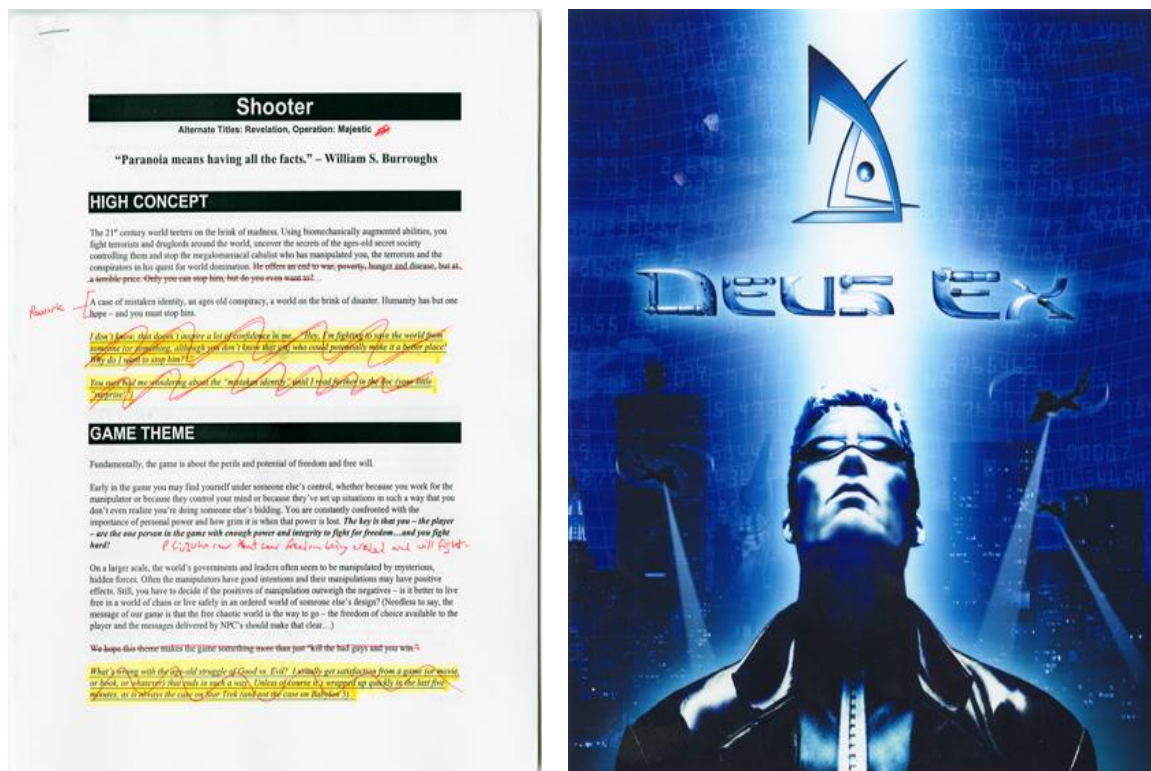


Figura 2.1: Documento do game *Shooter: Majestic Revelations*. O documento foi a primeira versão do game *Deus Ex* (à direita).

Game mechanics

Space combat

Goal: Space combat must be an exciting experience that is easy to get into so that it can serve as a "get in the door" feature for those not familiar with persistent world games.

Design: The space portion of the game takes place in algorithmically generated volumes of three-dimensional space. Each of these sections will be active and interesting, containing a wide range of stellar objects and server-controlled entities for players to interact with. Our intent is to give each section a "personality" that challenges players in many different ways. The graphics of the space environment will be vibrant, dynamic, and colorful. The ships and space-based facilities will have very different appearances, reflecting a wide range of different design priorities and maintenance conditions.

Space flight will be presented in the traditional first-person viewpoint. The level of immersion will be enhanced with a functional virtual cockpit, as well as a series of visual and audio cues designed to minimize the need to enter keyboard commands. Large ships will be controlled through a simple iconic interface capable of supporting several players performing different roles. Several automation features also allow a single player to easily control a large ship if the need arises. The highest level of accessibility will be our goal, and a joystick will not be required at any time to enjoy space flight.

Ship-ship combat will differ slightly from the established Wing Commander tradition via the inclusion of a maneuver energy play mechanic. This provides lower bandwidth internet-friendly play, while adding more strategy to the dogfighting experience. The offensive and defensive ship components are designed with Rock/Scissors/Paper elements. This allows players to balance the strengths and weaknesses of their ships to suit specific strategies while encouraging cooperation. We also foresee the large Capital Ships often being the focus of intense combat in a manner consistent with the Star Wars paradigm.

Planetary colonization

Goals: To provide a sense of ownership in the game. To formalize the process of building a player town in the wilderness and provide players a degree of governmental control over the cities they build. Hand in hand with housing, the goal of this feature is to increase the sense of ownership players have in the game, but beyond that, to provide an end point for the colonization process. It is also intended to add a political sub-game to PG.

Design: "Wild space" planets can be turned into members of Confed via colonization. Players have the ability to buy homes in various floor plans. These buildings are highly customizable, and cost a maintenance fee on an ongoing basis. A given character may only own one house. Once enough homesteaders are on a given planet, the village they heretofore formed is now eligible to become a city. Homesteaders gain (probably via their basic house interface) the ability to vote for

EA Confidential

Page 19 of 43

10/5/99

Figura 2.2: Fragmento do capítulo sobre mecânicas do documento de uma das versões do game

Wing Commander.

Encounter 6: Elevator/ Ground Option:

After clearing out the bad guys, the player will enter a cave that has a massive elevator (which won't be active until the bad guys are dead -- so no early escape for the player!).

The elevator will eventually bring the player to the top, but he'll have to use at least 3 separate ones to get there. At each stop the player will have to drive down corridors and up ramps to reach the other elevators, all while fighting more Covenant bad guys. Once the player reaches the ship, action continues in **Encounter 8**.

OR

If the player happened to steal a Banshee during Encounter 5, he can fly to the top (see **Encounter 7**).

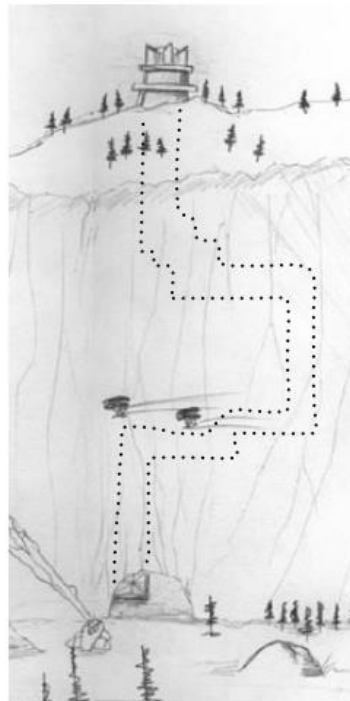


Figura 2.3: Fragmento do documento do game *Halo*. Cada missão do game foi escrita e ilustrada no documento.

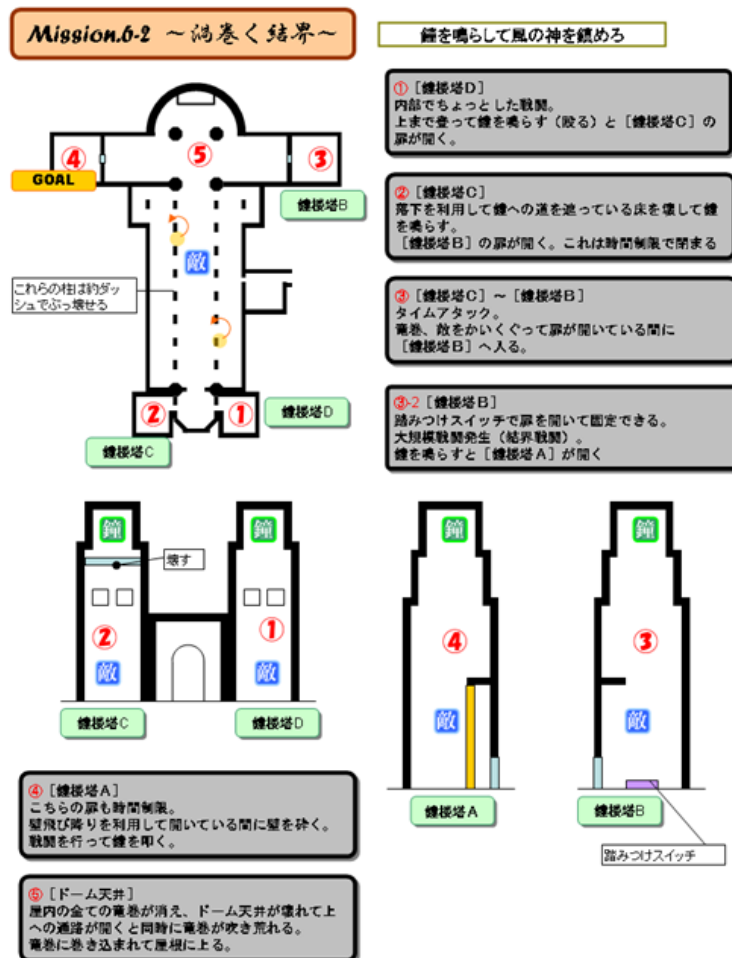


Figura 2.4: Fragmento do documento do game *Bayonetta*. Semelhante à *Halo*. O documento contém texto e ilustração para cada missão do game.

2.3 Críticas ao GDD

Apesar de largamente difundido e considerado imprescindível no processo de desenvolvimento, observamos que há uma pequena literatura que claramente começa a criticar o GDD, indicando que os mesmos levam muito tempo para serem criados, são difíceis de serem lidos, ocasionam erros de interpretação, são difíceis de atualizar e não possuem padrões dando margem a diversas interpretações acerca de seus conteúdos (que variam bastante) levando a falhas e retrabalho durante a produção [Lang 2009; Petrillo et al. 2009; Scheffield 2009; Souza 2009; Petrillo et al. 2008 Alves et al 2006; Callele et al 2005]. Tais críticas revelam que o GDD está longe de cumprir com o seu papel de conceituar, descrever e comunicar o que é o game a ser produzido, fazendo em muitos casos que a tarefa de extrair informações do mesmo para transformá-las em software seja,

por si só, um grande desafio [Callele et al. 2005]. Nesta seção incluímos alguns dos trabalhos que criticam ou relacionam problemas do desenvolvimento tendo como causa o GDD, na seção 2.3.1 apresentamos trabalhos que identificaram problemas analisando documentos de *postmortens*. Na seção 2.3.2 estão trabalhos nos quais relacionamos o GDD e os problemas ocasionados pelo mesmo e na seção 2.3.3 há um trabalho que relaciona a influência dos problemas do GDD como um dos fatores para aumento dos custos nos projetos.

2.3.1 Os Problemas da documentação identificados em estudos de documentos de *Game Postmortens*.

O trabalho de Sheffield [2009], publicado na *Game Developer Magazine*, analisou os *Game Postmortens*, documentos publicados ao final do desenvolvimento de um game que contém as lições aprendidas com o projeto (e são um dos poucos artefatos mantidos sem sigilo pela indústria). Basicamente, os *postmortens* analisados foram os publicados pelo *Gamasutra* e seguem o modelo proposto pelo *Open Letter Template*. Tal modelo se divide em três seções. A primeira delas trata do projeto em linhas gerais e apresenta alguns dos aspectos importantes do seu desenvolvimento. As duas seções seguintes discutem os aspectos que apontam o que foi bem sucedido ou não no projeto, e são intituladas respectivamente de “*O que deu certo*” e “*O que deu errado*” [Myllyaho et al. 2004]. No trabalho, que envolveu a pesquisa de diversos títulos lançados até o final da década passada, o autor apontou os dez erros mais comuns presentes no desenvolvimento de games. Um deles se referia diretamente a documentação, afirmando que a ausência de um artefato tecnicamente bem escrito, faz com que ocorra uma ruptura na comunicação entre os times de design e de implementação, gerando retrabalho para ambas as partes.

Analisando o mesmo objeto de estudo: os documentos de *Game Postmortens*, Petrillo realizou uma série de trabalhos [Petrillo et al 2008; 2009; 2010] com o objetivo de identificar um conjunto de fatores críticos e boas práticas que pudessem orientar futuras produções. Mais especificamente, o autor fez uma análise de vinte documentos, utilizando como único critério que tais documentos pertencessem a projetos de games finalizados. Os fatores críticos foram retiradas a partir de análises de discurso nas seções “*O que deu errado*”, já as boas práticas foram retiradas, usando-se o mesmo tipo de análise, porém nas seções “*O que deu certo*”. No Gráfico 2.1 podemos observar que

muitos dos erros são provenientes de decisões tomadas nas fases iniciais do projeto, como por exemplo: escopo fora da realidade, problemas de Design e ausência de documentação.

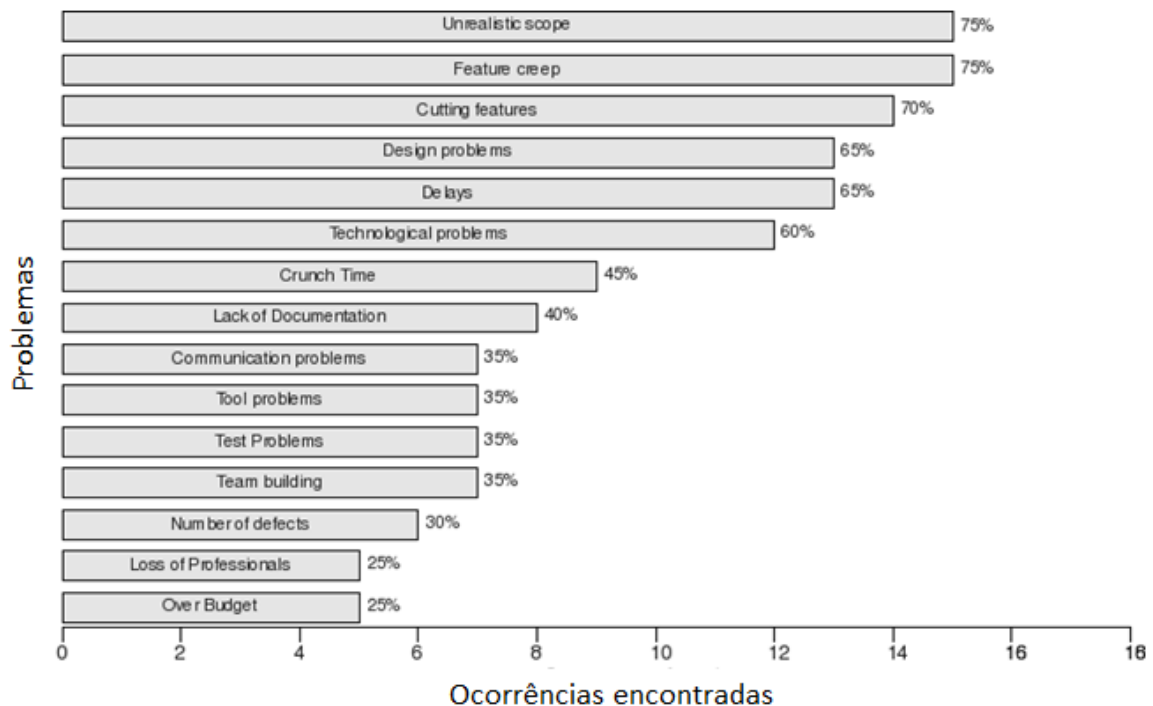


Gráfico 2.1: Lista de problemas e número de ocorrências dos mesmos [Petrillo et al 2009]

Alguns trechos dos documentos estudados ajudam a ilustrar o impacto que tais dados trouxeram às suas produções, como por exemplo as afirmações contidas no *postmortem* do Game **Wild 9** e no documento escrito por Barrett et al. [2002] respectivamente apresentadas abaixo.

“Muitas vezes, títulos atrasam porque há certos lapsos no design original que foram negligenciados e o design completo jamais esteve escrito no papel antes do desenvolvimento ser iniciado.”

“Pelo fato de estarmos projetando um Game para uma tecnologia (ao contrário do que se faz normalmente), nós nos livrávamos dos GDDs assim que eles estavam escritos. Os assets de Arte tinham de ser revisados, descartados e refeitos a partir do zero diversas vezes. Muitos leitores saberão a partir de suas experiências que esse é um cenário de features inadequadas, de ferramentas obsoletas e deadlines estourados.”

Considerando o fato de que muitos dos documentos analisados pertencem a games desenvolvidos em épocas distintas, podemos perceber que os erros identificados vêm se

repetindo ao longo dos anos. E se considerarmos que o trabalho de Petrillo analisa apenas *postmortens* de games finalizados, observamos que o problema é ainda mais abrangente do que se imagina, pois segundo resultados de pesquisas publicadas em [Isbister, K. and Schaffer, N. 2008] a quantidade de games finalizados e lançados no mercado reflete apenas cerca de 20% de todos os games que entraram em produção no mesmo período.

2.3.2 Problemas encontrados nos processos de desenvolvimento diretamente decorrentes da documentação

Um dos trabalhos mais citados nas pesquisas acerca de design e desenvolvimento de games [Callele et al. 2005], investigou o problema analisando o ponto de vista da engenharia de requisitos, seu trabalho indica o GDD como o artefato do qual os requisitos devem ser extraídos para que a equipe de desenvolvimento consiga planejar o game, de acordo com um conjunto de tarefas a serem concluídas dentro de um cronograma. O trabalho relata que o GDD possui inúmeras informações implícitas cuja análise deve ser cuidadosa e é sujeita a três tipos de implicações:

- Problemas derivados diretamente do material (o próprio GDD);
- Problemas derivados acerca do conhecimento e experiência relacionados ao desenvolvimento de games;
- E problemas derivados da tradução do conteúdo do GDD para as arquiteturas de software e hardware que executarão o Game finalizado.

O artigo relata que tais implicações podem resultar em esforços extras para o time de desenvolvimento além de provocar rejeição de ideias, o que resulta em impactos negativos para o processo criativo. Posteriormente, o próprio David Callele [Callele et al. 2006] aprofundou sua pesquisa com a proposta do Mapa de Intensidades Emocionais. Tal mapa foi a alternativa encontrada para identificar a reação do usuário diante de Requisitos Não Funcionais (RNFs), propostos pelo Design do Game, tais como medo e diversão. A dificuldade em conseguir identificar a efetividade de RNFs é complexa e a construção do Mapa de Intensidades Emocionais surge para atenuar esse fato e apresentar medidas concretas para as reações dos usuários. É uma forma para validar requisitos de design para posterior uso de técnicas que possam ser usadas pelos programadores para atingir tais RNFs. A pesquisa continuou a encontrar problemas nas especificações técnicas (GDDs entre outros) e um modelo de visualização foi construído a partir de protótipos de

games que usados em sessões testes avaliavam as reações dos jogadores. Dessa forma o autor revelou [Callele et al. 2009] que tal estudo ajudou na elaboração de técnicas que permitem projetar, construir e avaliar RNFs no desenvolvimento de games, diminuindo o conflito entre as fases de pré-produção e produção.

Um outro trabalho que discute a dificuldade de se obter requisitos a partir de GDDs foi apresentado por [Alves, C. et al. 2007], assim como nas discussões de Callele, a pesquisa aceita que a criação do GDD desempenha um papel crucial para as posteriores fases do desenvolvimento de games. Entretanto, não escondem as dificuldades de interpretação ocasionadas pela falta de padronização com que tais documentos são escritos.

Toda essa importância envolvendo a criação do GDD fica mais evidente quando analisamos a obra de John P. Flynt [2004]. Seu trabalho detalha todas as etapas do processo de desenvolvimento de software para games de forma muito semelhante aos descritos pelo *Rational Unified Process (RUP)*. Todas as fases: Concepção, Elaboração, Construção e Transição são revistas, ampliadas e atualizadas sob a ótica da produção de games, trazendo inclusive um conjunto de ferramentas e técnicas para facilitar o trabalho dos desenvolvedores. Entretanto, tal proposta só pode ser aproveitada em essência caso a fase de Concepção (do software) seja alimentada por informações precisas e corretas, no caso, as informações provenientes de artefatos como o GDD e derivados, o que, como já indicando por outros pesquisadores, está longe de acontecer.

Trabalhos como os de [Sommerville, I. And Sawyer, P 1997] e [Kujala, S. et al 2001] há muito têm revelado o quanto é custosa a negligência em se obter requisitos de forma precisa para as fases posteriores do desenvolvimento de software. E caso isso não ocorra como primeiro passo numa proposta como a citada anteriormente o risco de retrabalho será imenso. Na mesma linha de pesquisa, [Carvalho 2006] trata da criação de um processo preditivo de jogos para computadores pessoais. A grande inspiração para o modelo também foi o **RUP**. O interessante é que o autor cria a característica de orientação pelo Game Design, o que significa que todas as atividades do processo, para serem realizadas, devem ter um nível prévio de detalhamento presente no GDD. Além disso, há também o princípio de gerenciamento do GDD, que deve estar sempre sendo acompanhado e alinhado as necessidades de entretenimento do jogador. Nesse caso, o problema que reside aqui seria o da manutenção do GDD, algo que como exposto em [Machado 2009; Machado et. al 2012] não é tarefa das mais agradáveis pela equipe e

principalmente pelos responsáveis diretos pelo documento. No mais, a proposta deixa vaga até que ponto é preciso detalhar uma atividade no GDD para que a mesma possa ser realizada.

2.3.3 Problemas na documentação e sua influência nos custos dos projetos

De acordo com [Potanim, R. 2010] os problemas mencionados na seção anterior são alguns dos fatores que podem fazer com que a indústria de games venha a sofrer um crescimento não sustentável. Segundo seu estudo, realizado em empresas australianas de games, ele revela que a última década viu um crescimento exagerado da pressão das publicadoras que resultou no aumento dos custos de produção, na quebra de vários estúdios, no aumento de horas de trabalho dos desenvolvedores e na consequente diminuição da qualidade de vida dos mesmos.

Para se ter uma ideia, o trabalho relatou os custos de produção considerando o desenvolvimento de Games para as plataformas PC, Playstation 2 e Playstation 3 na Austrália. Em três anos (de 2004 a 2007) esse custo teve um aumento de aproximadamente 25% (de 12 para 15 milhões de dólares australianos) e nos três anos subsequentes (de 2007 a 2010) o aumento passou a ser ainda maior, considerando que o Game *L.A. Noire* [L.A. Noire 2011] custou em torno de 50 milhões de dólares americanos.

A pesquisa indica que a documentação e o cronograma estão sempre em conflito, principalmente porque as características dos projetos sempre objetivam estar atualizadas com as novas tecnologias. Segundo o autor esse conflito vem da observação de que normalmente aquilo que os designers pensam ser uma característica fácil de codificar, levam semanas para que um programador consiga terminar. E muitas vezes, tal trabalho é eliminado pelo produtor do game, pois este não concorda com os valores que tal característica pode agregar em relação ao produto final almejado. De acordo com a pesquisa, o cenário exposto revela os problemas de incompreensão acerca dos papéis e funções que a indústria australiana de games tem sido forçada a enfrentar, resultando em cronogramas irreais, muito retrabalho, várias horas extras e custos exagerados.

2.4 Nossas pesquisas preliminares: primeiro estudo junto à indústria

Em nossas pesquisas [Machado 2009, Machado et al 2010], observamos que muitos dos problemas apontados estão presentes na produção de qualquer tipo de game, seja ele casual ou grandes produções AAA. Elaboramos um questionário baseado no nosso estudo da literatura sobre dificuldades enfrentadas no cotidiano da produção de games e tal questionário foi respondido por 38 profissionais que trabalham no Brasil (34 do total) e no exterior (4 do total). Desses problemas, há um conjunto relacionados ao conflito entre as fases de pré-produção e produção do game que afligem, tanto pequenas quanto grandes empresas do setor e por isso tais dados podem ser analisados conjuntamente. Não se sabe onde exatamente se configuram os inícios e finais das fases de produção e os artefatos que deveriam esclarecer o projeto para o time, se revelaram insuficientes, principalmente o GDD. Por conta disso, muitos dos colaboradores da pesquisa responderam que o melhor é contar com a presença do Game Designer integralmente em todas as fases do projeto, já que os erros de interpretação são frequentes e muitos dos desenvolvedores não costumam ler os documentos [Gráfico 2.2], o que é um fato sustentado também pela entrevista do Designer Tim Lang [2009] ao *Game Carrer Guide*, na qual ele ironicamente responde que sua tarefa na indústria não consiste em programar ou gerenciar equipes, mas sim em “*escrever documentos que ninguém lê*”.

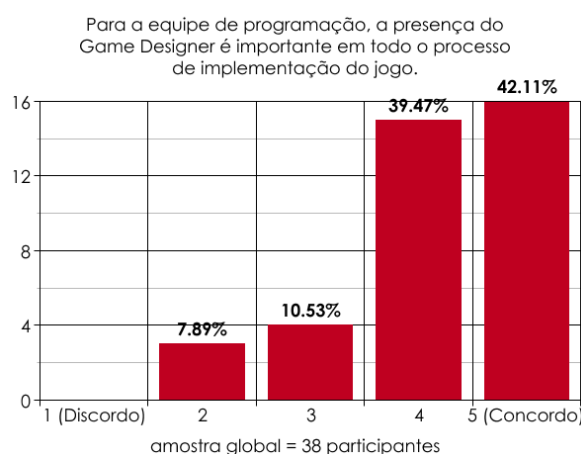


Gráfico 2.2: importância do game designer durante o processo de implementação.

Não é só em termos numéricos que essa necessidade se fez sentir, um dos participantes chegou inclusive a propor a criação de uma ferramenta aos moldes do que se encontra em softwares como o *Bugzilla* [Bugzilla, 2009], no entanto, adaptados as necessidades da produção de games. Nesse caso, a função de tal sistema seria a de registrar cada dúvida e entrega-las diretamente ao Game Designer. O problema de tal sugestão é que nem sempre o Game Designer terá a disponibilidade de atender a todos no tempo em que as dúvidas surgem e na forma com que se deseja, fazendo com que esse lapso (entre o registro e o atendimento da solicitação) acabe por criar atrasos, entre outros mal entendidos, que podem se propagar por diversas tarefas.

Falando em atendimento, os colaboradores da pesquisa também sugeriram que a criação de artefatos auxiliares ao GDD podem atrair mais usuários (leitores) e atenuar muitos problemas relacionados ao Design e a produção. Segundo os mesmos, o uso de máquinas de estados e protótipos são exemplos de recursos mais ricos e atrativos, que possuem um poder maior em explicar situações que o GDD, apenas, não consegue deixar claro [Gráfico 2.3].



Gráfico 2.3: Níveis de concordância em relação a geração de artefatos complementares ao GDD

Uma das preocupações acerca desses artefatos complementares, bem como do próprio GDD é a de que o Designer necessita definir uma estrutura adequada de apresentação para cada artefato, o que implica na escolha de ferramentas de documentação que sejam adequadas as atividades dos envolvidos. Assim sendo, é possível que a equipe de arte prefira receber seus artefatos em forma de arquivos de apresentação com imagens de referências em alta qualidade e riqueza de detalhes, contendo todas as informações possíveis sobre os significados e as técnicas utilizadas para a criação. Já a equipe de Programadores pode necessitar que os artefatos relacionados

a parte técnica sejam definidos como uma lista de requisitos contendo informações como a prioridade dos mesmos, os atores (jogadores ou sistemas) envolvidos e os conteúdos de arte associados (caso existam). Dessa forma, programadores, artistas e designers estariam alinhados com as mesmas ideias sobre o desenvolvimento do game. Em resumo, há aqui uma necessidade de adequar o GDD aos seus diferentes tipos de público.

Por fim, destacamos também a necessidade do GDD ser um documento mais propício a evoluir durante a produção do que a ter de estar totalmente pronto para que se comece a produção do game [Gráfico 2.4].

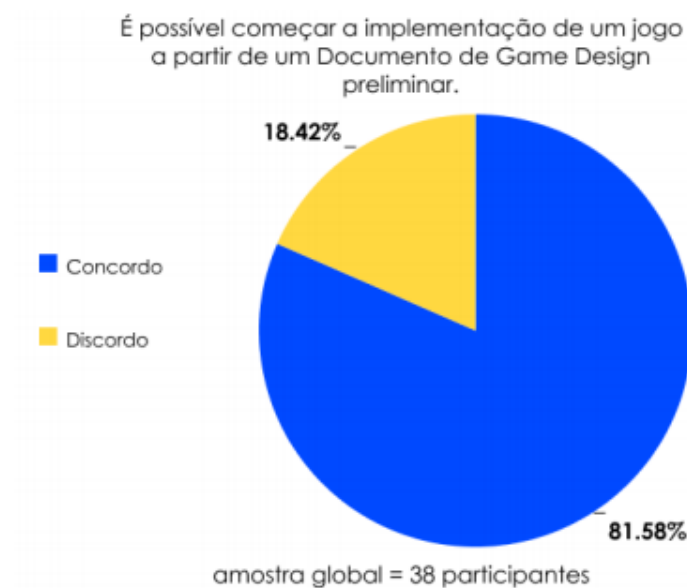


Gráfico 2.4: colaboradores concordam em iniciar a produção com um documento preliminar que evolui ao longo do processo

Inicialmente, perguntamos acerca da possibilidade de se iniciar a implementar um game a partir de um GDD preliminar e após isso, o que era mais fácil para a equipe de programação: implementar o game a partir de um documento finalizado ou de um preliminar com suporte do game designer. Nas respostas, conferimos que a maioria acredita ser mais adequado iniciar o desenvolvimento a partir de um conjunto inicial de definições do game e, que tal conjunto evolua de acordo com a evolução do projeto.

2.5 Diagnóstico: compilação de Inadequações e Conflitos do GDD

A partir do exposto neste capítulo e baseado nos resultados de nossas pesquisas e do estudo da literatura descritos anteriormente, conseguimos detalhar as inadequações e conflitos gerados pelo GDD, abaixo listamos o que até o momento entendíamos como um diagnóstico do problema.

2.3.1 Público – Alvo diversificado

O problema da incompreensão dos papéis e funções reflete um caso clássico de conflito multidisciplinar que observamos nos GDDs. Enquanto em muitos casos o mesmo documento é repassado para todos os membros do time, talvez o mais adequado seria pensar a sua criação e consequente distribuição para públicos distintos. A natureza multidisciplinar dos projetos de games faz com que as informações sejam muitas, mas nem todos os membros da equipe precisam consumir todas elas para que o projeto tenha um bom desenvolvimento, em outras palavras, talvez um artista não precise receber todos os detalhes acerca dos softwares que os programadores do game devem criar e vice-versa.

2.3.2 Diversidade de formas e conteúdos

Outro fator de inadequação diz respeito a forma e ao conteúdo do GDD, o tradicional documento de texto que intenciona explicar tudo nos mínimos detalhes tende a ser ignorado por muitos dos membros não justificando o esforço em criá-lo. Já o documento resumido traz lapsos que cria problemas de interpretações, bloqueios e atrasos na produção. A falta de padrões na forma e nos conteúdos descritos tornam o desafio de transformar as informações no game em um problema para o time de produção a cada documento diferente que se encontra. E por fim, é muito difícil saber exatamente com qual nível de detalhes se pode construir um GDD eficiente em passar sua mensagem para diferentes públicos sem cansá-los.

2.3.3 Imposição do GDD por culturas corporativas

É muito comum que o GDD mantenha as características das companhias e pessoas que o produzem. Como não há padrões definidos é normal que cada empresa e/ou time tenha as suas preferências acerca de como deve ser tal documento. O que significa que,

concordando ou não, os desenvolvedores têm de se habituar as exigências da corporação, excetuando-se os casos onde há a liberdade para o time decidir como deve ser o mesmo. No caso de uma imposição de formato, as equipes tendem a trata-lo como mero instrumento burocrático, como visto em [Winget & Sampson 2011] e atestado em nossas entrevistas, tornando o esforço em produzi-lo ainda mais desgastante para os times.

2.3.4 Evolução do GDD

Um dos conflitos mais comuns entre a pré-produção e a produção é a evolução do GDD. Membros da indústria revelaram claramente que a equipe atrasava a produção do game por conta dos lapsos no documento [Petrillo et al 2009], já que as características nunca estavam descritas suficientemente e faziam a produção tomar rumos incertos. Nas nossas pesquisas notamos um descompasso entre o game sendo desenvolvido e o game que foi documentado e nesses pontos não se sabe o que realmente precisa ser feito e quais direções seguir já que a orientação inicial do documento foi ignorada e a produção não representa (em parte ou no todo) o que foi definido na pré-produção. Uma das causas apontadas é a dificuldade em ter que atualizar o GDD, modificar características em um documento já escrito e depois retribuí-lo é tido como uma tarefa desgastante e muitas vezes resolvida melhor com uma consulta pessoal do que com uma alteração no documento. O problema é que essa consulta pessoal não tem a garantia de alcançar a todos no projeto.

2.3.5 O GDD e o Plano de Projeto

Outro conflito entre pré-produção e produção diz respeito a inadequação da conexão entre o Plano de Projeto e o GDD. Como o segundo contém as especificações do game, seria o material ideal para a construção de tal plano, mas a série de problemas de já mencionadas não fazem do GDD um bom candidato para tal tarefa.

Diante de todas essas inadequações e conflitos, chegamos a hipóteses para uma nova documentação de projetos de games baseada no nosso diagnóstico e que possam realmente ter sucesso na tarefa de guiar uma equipe de produção. As hipóteses são as seguintes:

- **Hipótese 1 (H1):** a documentação de games terá mais sucesso em guiar uma equipe de produção caso possa direcionar seu conteúdo a diferentes públicos alvo;
- **Hipótese 2 (H2):** a documentação de games terá mais sucesso em guiar uma equipe de produção caso dê suporte a vários tipos de formatos e conteúdos (texto, imagens, sons, vídeos, etc.);
- **Hipótese 3 (H3):** a documentação de games terá mais sucesso caso sua edição seja de controle do time e não de uma imposição corporativa;
- **Hipótese 4 (H4):** a documentação de games terá mais sucesso caso seja preliminar e iterativa durante a produção do que completa ao final da pré-produção;
- **Hipótese 5 (H5):** a documentação de games terá mais sucesso caso mantenha um bom diálogo com o plano de projeto e outros artefatos;
- **Hipótese 6 (H6):** a documentação de games terá mais sucesso em ser interpretada caso seja padronizada.

Para avaliar nossas hipóteses decidimos aprofundar o estudo através de entrevistas para entender, a partir da opinião de especialistas, quais as reais necessidades da indústria acerca do tema pesquisado e como podemos contribuir para a redução dos problemas.

Nota: as ilustrações de GDDs presentes neste capítulo estão disponíveis no [The UT Videogame Archive](http://www.cah.utexas.edu/projects/videogamearchive/media.php): <http://www.cah.utexas.edu/projects/videogamearchive/media.php>. (Universidade de Austin, Texas, EUA). Os fragmentos do GDD do game Bayonetta ®, podem ser conferidos em: <http://platinumgames.com/2009/12/02/the-ever-changing-game-design-of-bayonetta/>. Os fragmentos do GDD do game Halo ®, podem ser conferidos em: <http://halo.bungie.org/misc/HaloMissionbyRglass2002.pdf>

3 O problema do ponto de vista da indústria: entrevistas

Como resultado das nossas conclusões acerca de pesquisas anteriores e do que encontramos na literatura, decidimos iniciar um novo estudo, no qual pudéssemos aprofundar a discussão e entender, de acordo com o ponto de vista dos membros da indústria de games, os conflitos entre pré-produção e produção, que suspeita-se que seja, em grande parte, uma consequência direta do GDD [Callele et al 2005]. O objetivo foi avaliar nosso diagnóstico inicial testando as hipóteses levantadas para poder confirmá-las e descobrir se havia algum ponto não identificado que pudesse ser inserido para ter um diagnóstico mais completo. Para isso, fizemos contato com as empresas de games do Arranjo de Produção Local (APL) do Porto Digital e com profissionais que atuam em empresas internacionais. Nosso objetivo foi traçar um diagnóstico mais preciso, considerando o ponto de vista da indústria e que nos permitisse confirmar e entender seus principais problemas e necessidades, para propor uma solução baseada na visão dos nossos entrevistados. Para seleção dos participantes, nosso interesse teve atenção aos seguintes grupos específicos de desenvolvimento [Barros, 2010]:

- Grupo Administrativo - Compreende pessoas que lidam com a administração e alocação dos recursos de desenvolvimento. Além disso, são os papéis responsáveis pelas maiorias das decisões de um projeto de games.
 - Produtor/Gerente de Projetos;
 - Líder Técnico de Software/Game Designer;
- Grupo Operacional - São os encarregados de, a partir das decisões do Grupo Administrativo, transformar as idéias do projeto em produto, nesse caso, um game digital.
 - Programadores
 - Artistas
 - Designers

Em resumo realizamos entrevistas individuais na forma de estudos de caso múltiplo e holístico [Yin 2008] com integrantes dos grupos expostos acima com a

intenção de responder a questão: *O que a indústria de Games pode revelar acerca de suas necessidades para reduzir seus conflitos entre pré – produção e produção, decorrentes das documentações que as cercam?*

O quadro a seguir resume nossos objetivos com este estudo [Quadro 3.1].

Quadro 3.1: resumo dos objetivos de pesquisa

Analisar	As etapas de pré-produção e produção de games, bem como os artefatos produzidos e utilizados pelas mesmas;
Com a proposta de	Encontrar quais as necessidades que mais afligem os desenvolvedores (artistas, designers, programadores, produtores), principalmente em relação aos conflitos envolvendo a documentação, para propor uma ferramenta que dê suporte a atenuar os problemas relatados;
A partir do ponto de vista	Dos produtores, líderes técnicos, programadores e artistas;
No seguinte contexto	Produções de games dos participantes da pesquisa.
Tendo como unidade de análise	A produção de games;
Sendo um estudo de casos Múltiplos	Pelo fato de nossos entrevistados participarem da produção de diversos games;
E Holístico	Que nos permitiu estudar o contexto da pesquisa de forma global pelos vários pontos de vista envolvidos (designers, engenheiros, produtores, etc.)

3.1 Participantes do Estudo

Dentro dos grupos especificados, doze (12) participantes colaboraram com os resultados dessa pesquisa, todos são membros da indústria de games e trabalham em cinco empresas diferentes, tanto nacionais como internacionais. A idade média dos participantes ficou em torno dos vinte e nove (29) anos e os anos de experiência na indústria de games alcançou uma média de sete (7) anos, muitos dos entrevistados também adquiriram experiência de outras atividades na indústria de software, a maioria deles como programadores, designer de interfaces ou gerentes de projeto.

A questão da amostragem, naturalmente surge em pesquisas qualitativas e trabalhos mencionam que o importante é buscar a diversidade de casos e não a diversidade numérica de participantes [Flick 2004]. Para atender essa recomendação, buscamos entrevistar profissionais de diferentes empresas. Como, inicialmente, todas faziam parte do APL do Porto Digital, partimos em busca de outros desenvolvedores, com experiências externas a esse ambiente e que participaram de casos (produções) diferentes dos normalmente encontrados em Recife e no Brasil como um todo. Para expandir o nosso conjunto de casos, conseguimos participantes que atuam no Canadá e nos Estados Unidos. Terminamos com um total de 12 entrevistas, de acordo com [Yin 2008], não há um número fixo para se concluir essa fase da pesquisa, embora considere que a partir de 6 já é possível ter dados significativos e que o pesquisador deve estar atento a saturação (repetição) de respostas nas transcrições das entrevistas para considerar tal coleta de informações finalizadas. Devido a essas razões realizamos o objetivo de entrevistar membros que atuam nas seguintes disciplinas: Produção, Game Design, Artes e Programação. A quantidade e os papéis e dos participantes podem ser conferidos no Gráfico 3.1 a seguir.

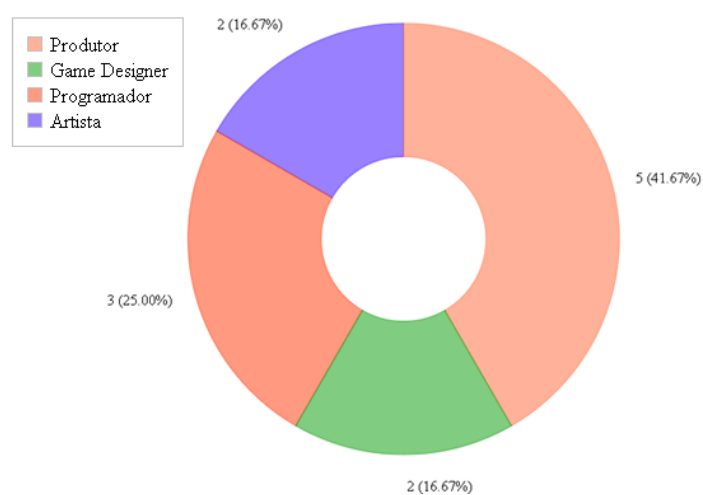


Gráfico 3.1: papéis dos participantes da pesquisa

Vale salientar que dos cinco (5) produtores que atenderam a pesquisa, dois (2) informaram que atuam também como Game Designers. Ainda segundo os mesmos, eles afirmaram que tal acúmulo de funções se deu pela experiência na área e que não se sentem sobrecarregados por este fato (ver Gráfico 3.2 na página seguinte). Além disso outros

participantes revelaram que começaram na indústria em cargos diferentes dos que ocupam hoje, alguns Produtores foram programadores anteriormente e alguns Game Designers já atuaram em tarefas ligadas as áreas de Artes ou Programação. Além disso, os Programadores e Artistas mais experientes, informaram ocupar cargos de liderança técnica nos projetos. Para evitar que a diversidade de funções e experiências afetasse as entrevistas, criamos várias formas de responder ao protocolo [ANEXO 1] que, junto ao método de entrevistas semi estruturadas, nos ajudou a evitar maiores complicações com tal fato.

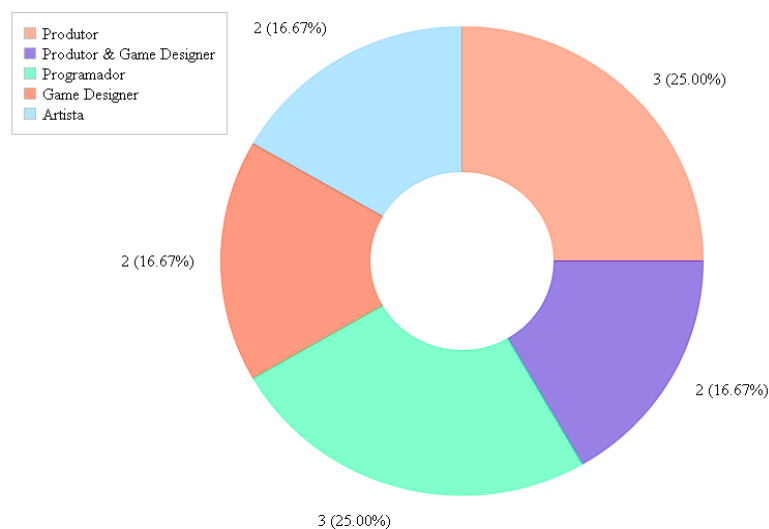


Gráfico 3.2: Papéis dos participantes da pesquisa (considerando Produtores que atuam como Game Designers).

Nessa configuração, Game Designers e Produtores nos forneceram informações acerca da concepção do game na pré-produção, da elaboração do GDD, entre outros artefatos, e como conduzem o processo nas fases subsequentes. Já os Programadores e Artistas nos revelaram como tais decisões da pré-produção afetam seu trabalho, como eles encaram o GDD e outros artefatos que lhes são entregues e como atuam junto aos Game Designers e Produtores para contornar os conflitos.

3.2 Entrevistas Semi Estruturadas

Com base nos trabalhos relacionados ao tema da pesquisa, apresentados no capítulo anterior, nós definimos um conjunto de assuntos para serem abordados nas nossas entrevistas (ver Anexo 1). As entrevistas foram respondidas por Produtores,

Engenheiros (Programadores), Designers e Artistas em seus próprios locais de trabalhos. Os entrevistados seguiram um *script* de questões, que foram conduzidas de forma livre para atender as possibilidades de conversação permitidas para a técnica como descrito por [Merriam 2009; Runeson & Host 2008]. Dessa forma, foi mais conveniente em situações nas quais um entrevistado podia responder questões referentes a papéis diferentes, como no caso dos participantes que atuam atualmente como Produtores e Game Designers. Estes estudos estiveram em prática por um mês. As sessões de entrevista apresentaram uma média de 20 minutos de duração. Nós utilizamos um gravador de áudio para registrar as sessões e prover meios de garantir uma transcrição mais rápida, segura e confiável. Ao início das entrevistas, todos os participantes foram informados de que os dados seriam sigilosos e que após a conclusão da pesquisa, seus registros em áudio seriam destruídos, garantido a segurança do anonimato de suas participações.

3.3 Procedimento de Análise: Análise Qualitativa de Conteúdo

Nosso procedimento de análise foi gerado a partir das transcrições das entrevistas discutidas na subseção anterior. Nós usamos a Codificação Teórica, que consiste em gerar conclusões de uma maneira clara e sistemática a partir dos dados coletados para serem codificados em categorias que ajudam o pesquisador a desenvolver suas bases teóricas [Runeson & Host 2008]. A Análise Qualitativa de Conteúdo é considerada como um procedimento clássico para se analisar materiais textuais que variam de produtos de mídia à produções textuais [Bauer & Gaskell 2000] e uma de suas principais características é a categorização, normalmente originada de métodos como a Codificação Teórica. Contrariando outras abordagens, o principal objetivo de tal análise é a redução do material [Flick 2004]. Tal proposição foi adequada a nossa pesquisa, pois nos ajudou no nosso processo de análise, já que precisávamos de um método rápido e fácil de gerenciar a quantidade de material textual produzida na transcrição das entrevistas. Nós seguimos um conjunto de etapas que foram definidas para tal proposta por [Mayring 2004], que consistia nas definições contidas na página seguinte.

Seleção do material	As entrevistas transcritas;
Análise da situação nas quais os dados foram coletados	Coleta realizada por meio de entrevistas, tendo como participantes profissionais da indústria de games, com o material sendo produzido através de transcrições do áudio dos entrevistados;
Direcionar a análise	A análise interpretou as entrevistas transcritas, procurando categorizar o discurso dos participantes de forma a atender os objetivos da pesquisa;
No seguinte contexto	APL do Porto Digital e indústria internacional;
Unidade de Análise	As entrevistas individuais de cada participante.

Desse modo, todas as nossas interpretações sobre os dados coletados foram categorizados em temas relevantes a nossa questão de pesquisa, os quais listamos abaixo:

- Pré-Produção do Game
- Documentos de Game Design
- Formas Alternativas de Documentar um Game
- Comunicação
- Protótipos
- Requisitos Fundamentais para produção de games
- Necessidades
- Ferramentas

Para nos auxiliar no processo de análise dos materiais textuais e montar as categorias, utilizamos a versão de avaliação do software ATLAS.ti [ATLAS ti 2012] (ver Figura 3.1). O programa fornece uma série de funcionalidades que facilitam a navegação nos materiais transcritos, a criação e edição de codificações, além de várias formas de visualização e consultas que permitem a localização rápida de palavras e citações no texto, bem como a criação de relações entre os mesmos.

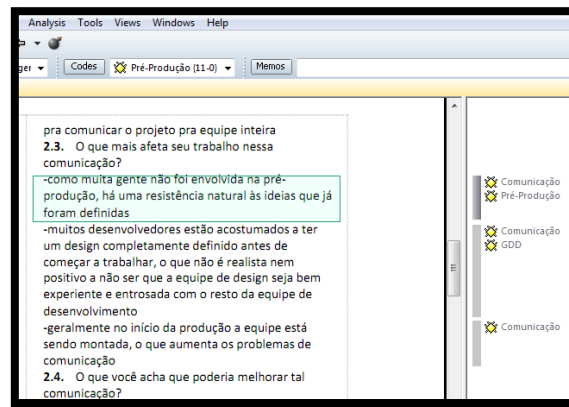


Figura 3.1: Interface do Atlas.ti apresentando o material textual (esquerda) e algumas codificações (direita) relacionada ao seu conteúdo.

3.4 Resultados das Entrevistas

Decidimos neste trabalho, categorizar [Flick 2004] o discurso dos entrevistados de acordo com os pontos discutidos em nosso primeiro diagnóstico. Abaixo colocamos as citações mais representativas e que se repetiram nas entrevistas acompanhadas da nossa análise acerca desses dados junto aos resultados do questionário (capítulo 2) e a nossa revisão literária. Nelas, revelamos nossas principais considerações, cada subseção reflete algum dos principais conflitos que os nossos participantes vivenciam em seus processos de desenvolvimento de games, bem como práticas adotadas para solução dos mesmos. O discurso dos participantes será apresentado entre aspas para diferenciar a opinião pessoal dos colaboradores de nossas considerações.

3.4.1 Público – Alvo do GDD

Em relação ao público alvo, os entrevistados foram claros em especificar três áreas como representantes dos profissionais envolvidos, embora menções ao Marketing e a Publicidade tenham aparecido em alguns textos dos participantes, Game Design, Arte e Programação foram sempre mencionadas como as principais disciplinas interessadas nos artefatos que cercam a produção do game, como atesta o comentário a seguir.

“O time se divide basicamente em três visões: Game Design, Arte e Programação. Então, nós necessitamos pensar nos elementos como representantes dessas três disciplinas... Aqui, nós dividimos o trabalho de acordo com essa visão...”

Tal opinião colabora com a hipótese (H1) de que o GDD pode ter maior sucesso caso possa ter conteúdo direcionado a públicos específicos como programadores, designers, artistas, entre outros da mesma forma como imaginou outro participante.

“...eu imagino que seria interessante uma ferramenta que você escreve o GDD e ela exporta para as pessoas que você quer.”

A hipótese também foi reforçada em outro caso, no qual o entrevistado refletiu sobre o GDD usado em seus projetos e suas questões para melhorá-lo, que lembram a necessidade de se direcionar o conteúdo para públicos específicos.

“Não tá perfeito do jeito que tá... estamos vendo que ferramentas a gente pode criar pra resolver esses problemas... de que formas pode ser escrito pra o artista? de que forma essa arte aparece no documento? O que podemos escrever para os programadores, etc.”

3.4.2 Forma e Conteúdo do GDD

Embora as empresas ainda usam o formato tradicional baseado em texto, vimos que muitas delas buscam por outras alternativas, como indica o comentário abaixo:

“Eu sinto a necessidade em termos de ter um GDD que seja mais prático, porque ele contém muito texto, então é fácil de esquecer algo. Talvez, eu acredito que pode ser mais interessante ter algo parecido com uma checklist.”

Já outras empresas, têm investido em outros formatos, como o apresentado abaixo, que parece ser bem conciso e objetivo e que não encontramos em nenhuma outra empresa participante da pesquisa e nem na literatura.

“...Nós usamos um GDD de uma única página apenas, para todos os módulos do game (...) nós definimos o que é necessário simplificar considerando a capacidade do time em alcançar as prioridades da produção, como data de lançamento, em determinados momentos.”

Em outro caso, um participante mencionou o uso de *mind-mappings* para auxiliar na documentação de elementos de programação.

“A equipe de programação costuma documentar as informações gerais em um mind-mapping quando estamos fazendo o gdd, organizando em nós o que é importante pra eles.”

GDD com *checklist*, com descrição de elementos em uma página e com *mind-mappings*, foram (entre outras) algumas das formas alternativas para documentar games apresentadas pelos participantes. Tal diversidade sugere a flexibilidade que o documento deve possuir em suportar diversas formas e conteúdos confirmando a hipótese 2.

3.4.3 Cultura Corporativa

A forma e o conteúdo sofrem influência da cultura corporativa da instituição, do projeto ou do próprio time como indica um dos participantes da pesquisa:

“Em um projeto maior, a gente usa um GDD (tradicional), até porque o cliente pede, então tem um pra ele (o cliente) e um pra equipe interna...”

Nesse caso os desenvolvedores têm um GDD para cumprir um requisito burocrático (o GDD do cliente) e um adotado pela equipe. É um caso no qual a própria cultura corporativa cria atividades extras levando os desenvolvedores a trabalhar em dois formatos (um que atende ao cliente) e outro de preferência do time, gerando um esforço desnecessário e corroborando para a hipótese (H3) de que a imposição de um documento para o time de desenvolvimento tende a gerar o insucesso do mesmo.

3.4.4 Evolução do GDD

Um dos grandes problemas citados, levou em conta a dificuldade de se manter o(s) artefato(s) atualizado(s). Segundo os entrevistados, a ideia de um documento completo que define tudo acerca do game é equivocada como atesta o colaborador no comentário abaixo:

“Eu acho que o GDD não é o único processo que tem a essência do jogo... na verdade a essência do jogo deve estar na cabeça do time. Eu digo isso, porque eu vejo uma preocupação muito grande com o documento de game design, todo mundo tenta deixar ele bem fechado antes de começar a produção e um documento de game design nunca está fechado, eu digo sempre isso em aulas, em palestras, pra times que estão começando... É muito comum com o tempo a gente desenvolver 10 features em cima da feature original que o cliente propôs, então, pra mim, o registro de um design de game deve ser feito em post-it ou em papel porque são coisas voláteis, fáceis de manipular... É registrar suas ideias e saber pra onde ir... porque game é sobretudo iteração... então você faz, testa, faz de novo, vê se ficou bom e vai.”

Casos como esse, foram informados pelos participantes mais experientes, e tratam da intensidade com que as mudanças ocorrem no desenvolvimento de games, o que exige uma flexibilidade na(s) documentações, que pouco existe(m) no(s) formato(s) clássico(s). Para isso, os participantes têm investido em meios alternativos que atendam tais necessidades, como o já citado na subseção 3.4.2 deste capítulo e no comentário abaixo:

“Nós estamos usando um modelo, chamado Game Design Logs, ele foi criado para dar suporte ao formato ‘vivo’ de um GDD e tem sido muito produtivo para nossos projetos. Toda a documentação é feita por pequenos registros, com poucos detalhes e compartilhados com todo o time, todos podem ver e fazer mudanças em tempo real ou pelos registros que ficam sempre salvos. Então, isso tem sido muito simples de trabalhar e atendeu muitas de nossas necessidades, principalmente a de manter um GDD... Antes disso o design e o game (em desenvolvimento) eram sempre assíncronos.”

O comentário acima confirma a hipótese (H4) de que o GDD não precisa ser completo antes de se iniciar a produção e sobretudo deve se manter simples para ser atualizado sempre que existir a necessidade para tal.

3.4.5 O GDD e o Plano de Projeto

A intensidade de mudanças e a dificuldade de manter o GDD atualizado revelam como é próxima a relação entre o artefato e o planejamento do projeto. A quantidade de iterações presentes nos processos atuais foi considerada um dos fatores de se ter de mudar as especificações do game o tempo todo, como no comentário logo abaixo:

“O principal problema é a quantidade de iterações que temos de fazer durante nosso processo... Basicamente, você tem de fazer mudanças o tempo todo...”

O que fica mais evidente caso tais especificações nunca tenham sido bem definidas em um primeiro momento, como atesta o comentário de outro participante:

“Todas as definições da pré-produção são importantes e ajudam na produção. Mesmo com elas sendo iteradas durante o projeto, ter um ponto de partida e um objetivo final é imprescindível”

Os comentários acima reforçam a hipótese e a necessidade de se ter um documento flexível a mudanças (H4) e que possa explicar a cada iteração os objetivos a serem alcançados mantendo um bom diálogo com o plano de projeto e outros artefatos da

produção (H5), caso contrário problemas entre o design e o desenvolvimento podem ser acumulados durante o projeto.

3.4.6 Game como Documento

Um dos participantes, membro de uma tradicional empresa reconhecida por seus títulos AAA, revelou ter parado de investir em documentações e ter passado a usar protótipos que, segundo o mesmo, possuem um poder de expressão maior para explicar como o game deve ser implementado:

“Nós paramos de criar documentos para explicar como o game deveria ser. Nós agora desenvolvemos protótipos e mostramos como os games devem ser! Nós reduzimos nosso processo de documentação e deixamos as coisas bem simples de serem especificadas e concluídas.”

No entanto, apesar do sucesso citado pelo entrevistado anterior, há quem não concorde com a diminuição drástica da documentação (sendo este, mais um entre tantos conflitos nos quais o GDD está inserido):

“Alguns encaram como trabalho desnecessário visto que eles “sabem o que estão fazendo”. Entretanto esta documentação é importante para melhorar a chance que a estrutura técnica do jogo e suas partes mais desafiantes sejam implementadas com sucesso.”

Vimos que mesmo com uma discussão mais ampla, que procurou abordar a visão dos profissionais acerca dos problemas que cercam a Pré-produção e a Produção de games, o GDD, pelo que foi dito por nossos entrevistados, continua a ter importância contraditória e está no coração de muitos dos conflitos apresentados. Tal contrariedade se reflete principalmente porque a função de guiar a equipe durante a dinâmica produção de um game continua a não ser atendida. Revisitando o diagnóstico, encontramos nas entrevistas muitos dados que o reforçam e confirmam várias de nossas hipóteses e que nos permitiram incorporar um novo elemento na discussão: a possibilidade das versões do game (protótipos) servirem como seu próprio documento (subseção 3.4.6 deste capítulo). Além disso, recusamos outras hipóteses (especificamente a H6) por conta de comentários como o apresentado abaixo que desconsidera a padronização do GDD:

“Você tem que entender que cada (projeto de) game é uma história diferente e isso não dá pra ser padronizado”

Tal opinião se repetiu em outras formas durante as entrevistas e nos revelaram que a padronização do GDD, pelos resultados que obtivemos é um falso problema. Cada história (projeto) é diferente e precisa dos meios corretos para ser contada, padronizar isso é como forçar que todas as histórias sejam sempre contadas do mesmo jeito.

3.5 Conclusões sobre os resultados das entrevistas

Além de termos avaliado nossas hipóteses, notamos que os resultados das entrevistas nos forneceram um material rico que expôs muitos conflitos da pré – produção e que merecem atenção especial nesta subseção.

3.5.1 Investimento em pré–produção

Pudemos observar que o tema Pré-Produção gerou muita discussão durante as entrevistas. Notamos pelas respostas dos usuários que o principal objetivo de tal fase é o desenvolvimento do *Game Concept*. Isso foi citado por profissionais de pequenas e grandes empresas e até uma surpresa considerando que a Academia cita constantemente o esforço feito pelas empresas para produzir GDDs, tendo-os como o principal artefato da Pré-Produção (Capítulo 2). Um dos participantes, membro de uma empresa que já produziu diversos títulos AAA, declarou que seu time de desenvolvimento precisa de três a nove meses de Pré-Produção, na qual uma das principais tarefas é a definição do *Game Concept*. Muitos dos outros participantes da pesquisa responderam que o *Game Concept* é uma das principais atividades durante Pré-Produção, mas se mostraram preocupados com o investimento feito nessa fase, principalmente em relação ao tempo, considerado insuficiente.

3.5.2 Gerenciamento das características do game

Como observado na seção 3.4.5, participantes mencionaram o número de iterações como um problema em seus processos. De fato, os métodos ágeis usados por todas as empresas visitadas permitem essa flexibilidade [GameDev 2009]. Entretanto, como nossos entrevistados disseram, a quantidade de mudanças é tão intensa que implica em uma dificuldade na gerencia das características do game. Em muitos casos, as mudanças

afetam as três principais disciplinas (Design, Programação e Artes) e por conta da grande quantidade de detalhes precisam ser discutidas e repensadas iteração após iteração, o que se torna perigoso a medida que o escopo do projeto aumenta.

3.5.3 Código versus documento

Uma importante discussão surgiu a partir da entrevista de um dos colaboradores que trabalha em uma grande empresa do setor, especialista no desenvolvimento de títulos AAA de sucesso. A discussão diz respeito ao código do game servir como documento e, se o mesmo pode vir a substituir o GDD. A implementação de protótipos, ou partes funcionais do game tornam-se ferramentas com poder de expressão melhor para comunicar o time de desenvolvimento o que deve ser construído. A documentação do game torna-se o game em si. E isso pode se tornar em um grande aliado a evitar os lapsos dos tradicionais GDDs e além disso fornecer melhores resultados em relação ao que é “documentado”. Nesse cenário, podemos pensar questões como: Quais as consequências de não ter uma documentação tradicional? Tal abordagem é realmente benéfica? Como as mudanças no projeto serão comunicadas ao time? E se acontecer mudanças na equipe, como os novos membros entenderão o que deve ser feito?

3.5.4 Academia versus indústria

Durante as entrevistas e a fase de análise observamos divergências entre Academia e Indústria. O que é visto como necessário pela academia não é encarado pela indústria com a mesma relevância e um dos exemplos é o próprio GDD. Enquanto um bom número de textos propõe formatos de criar e adaptar o GDD a diferentes situações, muitos dos entrevistados declararam que o uso do GDD em seus processos é praticamente inexistente e o principal artefato criado na fase de Pré-Produção é o *Game Concept*. O que nos levou a pensar se o GDD como proposto pela academia é realmente necessário. Falando em Pré-Produção, outra divergência é a ideia que a Academia traz de uma fase de Pré-Produção capaz de descrever cada detalhe do game a ser desenvolvido de maneira suficiente a guiar o time durante a Produção. Isto não foi compartilhado pelos entrevistados. Pelas práticas adotadas, tal visão é considerada cara e pouco realista.

É bom deixar claro, que os mesmos não descartam a importância de tal fase, mas a realizam de forma a poder chegar mais rápido nos resultados (em muitos casos protótipos) para poder efetuar avaliações e melhorá-los ao longo do processo. Muito mais

do que uma divisão clássica de pré-produção, produção e pós-Produção, o que percebemos com os nossos resultados é que a indústria parece ter adotado ou está se inclinando a aceitar um modo de desenvolvimento no qual, a partir de uma primeira pré-produção, na qual os principais detalhes e a visão do projeto são construídos, entra-se em vários ciclos de pré-produção, produção e pós-produção no qual o game é sempre avaliado, modificado e melhorado de acordo com os resultados apresentados e com as capacidades técnicas e orçamentárias da equipe e do projeto. Nesse cenário extremamente dinâmico, os modelos clássicos de documentação baseados em longas explicações textuais tornam-se inconvenientes e as formas alternativas esbarram nas preferências pessoais dos membros das equipes, nas equipes como um todo e nas particularidades do projeto.

4 Princípios para uma nova documentação de games

Nós consideramos que os resultados obtidos na revisão da literatura, no *survey* que fizemos preliminarmente e nas entrevistas (capítulo 3), servem de diagnóstico preciso dos atuais desafios e práticas adotadas pelos desenvolvedores de games em termos de documentação, abrindo espaço para a proposição de novas ferramentas de apoio. Nas próximas sessões apresentaremos nossas principais considerações acerca de como todos esses resultados influenciaram no desenvolvimento de princípios para construção de uma ferramenta que visa atenuar os problemas relatados.

4.1 Novo diagnóstico: atualizando a compilação de inadequações e conflitos do GDD

Como revelado no capítulo 3, vimos que mesmo com uma discussão mais ampla, na qual o GDD foi abordado de diversas maneiras, acreditamos, pelo que foi dito por nossos entrevistados, que continua a ser contraditória sua importância em todo o processo. Contrariedade que se deve, principalmente, ao não cumprimento da função de guiar a equipe durante a dinâmica produção de um game. Sendo assim, revisitamos os conflitos e inadequações investigados no projeto, e a partir das discussões deste capítulo atualizamos as considerações ao nosso primeiro diagnóstico em relação ao resultado das entrevistas. Acreditamos que em decorrência do que foi exposto, os pontos de inadequação e conflitos presentes, devem ser encarados como princípios para atenuar problemas da Pré-Produção e Produção de games, considerando o uso de um artefato, o GDD, porém atualizado e corrigido de acordo com o ponto de vista dos profissionais entrevistados.

4.1.1 Público – Alvo diversificado

- É necessário que qualquer envolvido na produção possa criar informação usando os recursos de sua preferência, de maneira rápida e fácil.
- É necessário que as informações, uma vez criadas, sejam acessadas facilmente por qualquer envolvido na produção para que se facilite o trabalho em equipe e a colaboração na construção de ideias e da visão unificada do projeto.
- É necessário que as informações sejam direcionadas de acordo com o público alvo do documento (programador, designer, artista, etc.);

4.1.2 Diversidade de formas e conteúdos

- É necessário que a documentação em projetos de games seja flexível quanto a sua forma e conteúdo para atender melhor a públicos multidisciplinares.
- É necessário que qualquer tipo de conteúdo (mind mappings, post its, ilustrações, etc.) possa ser armazenado.
- É necessário que o sistema permita visualizações sobre os diferentes conteúdos armazenados.

4.1.3 Imposição do GDD por culturas corporativas

- É necessário que a documentação em projetos de games possa se adaptar facilmente as Culturas Corporativas das instituições e times independentes que desejem usá-la.

4.1.4 Evolução do GDD

- É necessário que a documentação em projetos de games possa se adequar facilmente as inúmeras mudanças que ocorrem durante as iterações do projeto.
- É necessário que a documentação possua uma estrutura que seja simples de acessar e alterar em qualquer momento do projeto ou sempre que um dos envolvidos na produção tenha tal necessidade.

4.1.5 A Documentação e o Plano de Projeto

- É necessário que a documentação do game permita que a qualquer momento do projeto, a equipe possa consultá-la e usá-la como instrumento para alimentar os planos de projeto em etapas como elicitação de requisitos ou criação e distribuição de tarefas.

4.1.6 O GDD como ferramenta para gestão de conhecimento

- É necessário que a documentação do game permita que informações e decisões de projeto sejam facilmente rastreadas.

4.1.7 Código como documento

É necessário que a documentação permita que as instâncias do código que rodam o game (ou seus protótipos) sejam facilmente consultadas e executadas a partir da mesma por qualquer envolvido na produção.

A seguir, começaremos a detalhar a implementação de uma ferramenta, um artefato “vivo” que funcionará como uma plataforma de conversação, cujo objetivo é materializar o discurso dos participantes da pesquisa na busca de meios de diminuir os problemas entre as fases no processo de desenvolvimento de games.

4.2 Inspirações

Retiramos da literatura algumas lições sobre como apresentar GDDs. A diversidade é grande, mas nesta subseção apresentamos trabalhos que trazem tanto as vantagens quanto desvantagens das técnicas relatadas.

Alguns autores [Rollings & Morris 1999] defendem que um GDD possui a característica de nunca estar completo, ser interativo e dinâmico. Por isso mesmo, eles propõem que o melhor formato de um GDD é o de uma página *Wiki*, pois pode ser acessado e editado de forma colaborativa, possui controle de versão e suporta múltiplos acessos. As críticas [Lang 2009] a essa proposta são o fato de que, quem edita precisa conhecer linguagens de *tags* e outros tipos de marcações, além das dificuldades em imprimir todos os conteúdos formatados pelo *HTML* e a perda de foco que pode ser causada por uma edição carregada de *hiperlinks*.

Em outro trabalho foi investigado o uso de multimídia [Souza 2009] como uma forma alternativa de documentar projetos de games. Inicialmente sua pesquisa teve o objetivo de identificar quais formatos de comunicação eram mais adequados para se apresentar às equipes. Os formatos considerados foram: textos, imagens e animações. Todos foram descritos em uma matriz morfológica considerando os pontos positivos e negativos de cada um dos formatos analisados [Tabela 4.1]. Embora o autor tenha tentado identificar qual formato se apresentaria mais adequado ao processo, a ideia de fornecer documentos em vários formatos diferentes é atrativa, pois tal diversidade pode alcançar

a compreensão de mais membros da equipe. No entanto, tal alternativa pode aumentar os custos do projeto, sobretudo em fases preliminares, mas pode justificar o investimento por aumentar a compreensão do time nas fases subsequentes do processo.

Tabela 4.1: Dados da pesquisa de [Souza 2009] indicando as vantagens e desvantagens no uso de três formatos para elaboração de GDDs.

	TEMPO DE PRODUÇÃO	PONTOS POSITIVOS	PONTOS NEGATIVOS	OPINIÕES
TEXTO	1 hora	- Protótipos próximos do concebido	- Nenhum tinha mecânica igual ao concebido - Pontos com falta de detalhamento	- Ideal para levantamento de requisitos - Difícil interpretação
IMAGEM	30 minutos	- Protótipos visualmente parecidos - Um protótipo tinha mecânica e regras exatamente como concebido	- Variação nas regras	- Fácil entendimento - Simples e rápida
ANIMAÇÃO	2 horas	- Teoricamente forma ideal	- Nenhum protótipo sequer parecido - Regras de difícil entendimento - Muita variação visual	- Esforço de produção talvez não compensasse

Uma outra alternativa interessante aos formatos é a proposta apresentada por Daniel Cook [Cook, D. 2011]. Em seu processo de desenvolvimento é usada uma técnica chamada de *Game Design Logs*. A ideia é a de construir o documento a partir de pequenos *Logs*, ou seja, registros que informem ao time de desenvolvimento qual a ideia a ser desenvolvida ou melhorada em determinado momento da produção. Cada *Log* quando publicado contém um conjunto de informações que indicam o status do desenvolvimento do game, a reação do time a esse desenvolvimento e o que fazer para solucionar os problemas já encontrados. Toda publicação de um novo *Log*, se sobrepõe ao anterior para que o documento sempre fique atualizado, como se fosse uma lista de discussão presentes nos clientes de e-mails. Dessa forma, o *Game Design Logs* intenciona facilitar a tarefa de criar, manter e ter sempre atualizados os registros das decisões do projeto.

As críticas a tal abordagem, presentes na postagem do próprio autor, é que ela no momento se limita a comentários em textos e hiperlinks, quando seria interessante lançar mais recursos como uma forma de organizar e recuperar *logs*, além de uploads de arquivos diversos, incluindo protótipos do próprio game em desenvolvimento. Se os *logs* (e o próprio GDD) têm a função de registrar e comunicar as ideias do game à equipe de desenvolvimento, então por que os *logs* e os artefatos produzidos continuam separados?

4.3 Requisitos

A inspiração para construção do protótipo teve como base o método de mapear necessidades de usuários em requisitos de sistemas, proposto em [Kujala, S. et al. 2001] e já aplicado em um bom número de pesquisas como [Falcão & Gomes 2004]. Usamos tal método, inicialmente, como uma das referências para condução das entrevistas e posteriormente na implementação. O trabalho investigou o envolvimento de usuários nas fases iniciais do desenvolvimento de produtos. Como resultado, além da constatação dos ganhos de satisfação e usabilidade, obtidos a partir de uma extensa revisão literária, a autora desenvolveu um método para tornar a descoberta de requisitos, mais segura e efetiva em termos de custo.

Inicialmente, identifica-se quais as reais necessidades dos usuários para depois traduzi-las em Requisitos. Como nossos princípios são todos baseados nas necessidades

observadas no resultado das nossas pesquisas, nós os usamos para realizar o mapeamento. Particularmente, nosso interesse em tal abordagem para a etapa de prototipação aconteceu devido ao fato do trabalho mencionar que “*o envolvimento de usuários nas fases iniciais é útil mesmo que aconteça em um pequeno espaço de tempo e em projetos produzidos sob baixo orçamento*” como uma das principais implicações positivas do método. Isso porque as características de “*pequeno espaço de tempo*” e “*baixo orçamento*” foram características deste trabalho. Abaixo, apresentamos [Tabelas 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7] o mapeamento dos princípios em requisitos de sistema.

Tabela 4.2: Mapeamento de princípios e requisitos para o diagnóstico acerca do público alvo.

Princípios	Requisitos
<ul style="list-style-type: none"> · É necessário que qualquer envolvido na produção possa criar informação usando os recursos de sua preferência, de maneira rápida e fácil. 	<ul style="list-style-type: none"> • Permitir que usuários postem informações sempre que acharem necessário. • Permitir que usuários possam fazer uploads de qualquer tipo de arquivo • Permitir fácil acesso as informações presentes no sistema
<ul style="list-style-type: none"> · É necessário que as informações, uma vez criadas, sejam acessadas facilmente por qualquer envolvido na produção para que se facilite o trabalho em equipe e a colaboração na construção de ideias e da visão unificada do projeto. 	<ul style="list-style-type: none"> • Permitir que usuários possam trabalhar de forma colaborativa • Permitir que usuários possam comentar as postagens. • Permitir que qualquer informação possa ser direcionada a um ou mais públicos específicos.

Tabela 4.3: Mapeamento de princípios e requisitos para o diagnóstico acerca de formas e conteúdo.

Princípios	Requisitos
<p>· É necessário que a documentação em projetos de games seja flexível quanto a sua forma e conteúdo para atender melhor a públicos multidisciplinares.</p>	<ul style="list-style-type: none"> • Permitir que o sistema armazene diversos tipos de arquivos; • Permitir que o sistema gere visualização sobre diversos tipos de arquivos; • Permitir que o sistema permita fácil navegação no seu conteúdo

Tabela 4.4: Mapeamento de princípios e requisitos para o diagnóstico acerca de culturas corporativas.

Princípios	Requisitos
<p>· É necessário que a documentação em projetos de games possa se adaptar facilmente as Culturas Corporativas das instituições e times independentes que desejem usá-la.</p>	<ul style="list-style-type: none"> • Impedir que o sistema seja intrusivo; • Permitir que o sistema tenha fácil aprendizado; • Permitir que as funcionalidades do sistema sejam rapidamente executadas.

Tabela 4.5: Mapeamento de princípios e requisitos para o diagnóstico acerca da evolução do GDD.

Princípios	Requisitos
<ul style="list-style-type: none"> · É necessário que a documentação em projetos de games possa se adequar facilmente as inúmeras mudanças que ocorrem durante as iterações do projeto. 	<ul style="list-style-type: none"> • O sistema deve estar sempre disponível; • O sistema deve ser simples de ser modificado;
<ul style="list-style-type: none"> · É necessário que a documentação possua uma estrutura que seja simples de acessar e alterar em qualquer momento do projeto ou sempre que um dos envolvidos na produção tenha tal necessidade. 	

Tabela 4.6: Mapeamento de princípios e requisitos para o diagnóstico acerca do diálogo do GDD com o plano de projetos.

Princípios	Requisitos
<ul style="list-style-type: none"> · É necessário que a documentação do game permita que a qualquer momento do projeto, a equipe possa consultá-la e usá-la como instrumento para alimentar os planos de projeto em etapas como elicitação de requisitos ou criação e distribuição de tarefas. 	<ul style="list-style-type: none"> • Integração com ferramentas de gerenciamento de projetos; • Geração de relatórios sumarizados.

Tabela 4.7: Mapeamento de princípios e requisitos para o diagnóstico código como documento.

Princípios	Requisitos
<ul style="list-style-type: none"> • É necessário que a documentação permita que as instâncias do código que rodam o game (ou seus protótipos) sejam facilmente consultadas e executadas a partir da mesma por qualquer envolvido na produção. 	<ul style="list-style-type: none"> • Permitir que os usuários possam armazenar qualquer instância do game em produção • Permitir que as instâncias do game sejam facilmente encontradas • Permitir que os usuários possam executar as instâncias do game através do sistema

4.4 Classificação e análise dos Requisitos

Como apresentado anteriormente, o mapeamento dos princípios foram resultante da nossa análise dos dados provenientes da pesquisa. A partir deles, todos obtidos a partir de necessidades apontadas na fase de entrevista e análise, identificamos um conjunto de requisitos que serviram de base para construção do protótipo. A seguir apresentamos uma classificação básica dos mesmos, bem como cada um deles se encaixa no direcionamento do discurso dos entrevistados.

4.4.1 Requisitos Funcionais

- **[RF 001] Permitir que usuários postem informações sempre que acharem necessário:**
Esse Requisito relaciona-se ao fato do GDD ser considerado um artefato vivo, sempre sujeito a mudanças, como os entrevistados mencionaram. Um artefato desse tipo deve permitir que seus usuários postem informações (um texto, uma imagem, um vídeo, etc.) sempre que achar necessário.

- **[RF 002] Permitir que usuários possam fazer uploads de qualquer tipo de arquivo:**

Como a informação em um GDD não deve se limitar a texto, como os entrevistados e nossas revisões de literatura sugerem, o sistema deve permitir o upload de qualquer tipo de arquivo. Não se sabe o que pode explicar melhor uma determinada característica do game, pode ser uma foto, um som, um texto ou até mesmo um game. Então, o sistema deve estar apto a permitir tal diversidade.

- **[RF 003] Permitir que o sistema armazene diversos tipos de arquivos:**

Como o sistema deve garantir diversidade de tipos de arquivos que os usuários podem disponibilizar, o mesmo deve trabalhar com um modelo de persistência adequado para tal armazenamento e além disso, ter a escalabilidade de se adaptar a diferentes características de diferentes projetos. Por exemplo: Armazenar os *assets* de um game casual talvez não necessite de tanto espaço quanto o de uma produção AAA.

- **[RF 004] Permitir que o sistema tenha visualização sobre diversos tipos de arquivos:**

Devido a necessidade dos usuários trabalharem com uma grande quantidade de arquivos diversos é necessário que o sistema garanta uma forma de visualizá-los. Sabemos que o público alvo do GDD é distinto e têm várias preferências quanto as formas de editar tal artefato, esse requisito, assim como os requisitos **[RF 002]** e **[RF 003]** estão diretamente relacionados a atender a diversidade de público do GDD e ao conteúdo e forma que tal público achar mais adequado.

- **[RF 005] Permitir que usuários possam trabalhar de forma colaborativa:**

A colaboração permite que os envolvidos na produção possam compartilhar a mesma visão de design do game, além disso, como os problemas de comunicação são frequentes nesse tipo de produção, o uso colaborativo se faz mais do que necessário. Dessa forma, todos têm acesso direto ao documento e suas contribuições ou questionamentos serão disponibilizadas a todos.

- **[RF 006] Permitir que usuários possam comentar as postagens:**

Diretamente relacionado ao requisito anterior está o fato dos usuários poderem comentar as postagens, dessa forma, problemas de comunicação podem ser atenuados e todas as dúvidas e contribuições estarão associadas a postagem original.

- **[RF 007] Permitir que os usuários possam armazenar qualquer instância do game em produção:**

Em alguns casos, vimos que o game em si pode ser seu próprio documento. Sendo assim, os usuários produzem e avaliam vários protótipos (instâncias do game) em ciclos iterativos, os quais o sistema deve armazenar.

- **[RF 008] Permitir que as instâncias do game sejam facilmente encontradas:**

A possibilidade de poder armazenar vários protótipos faz com que surja a necessidade de poder localizá-los rapidamente. Normalmente os colaboradores que participaram da entrevista, usam metodologias ágeis e baseiam sua produção em protótipos e avaliações, como vimos na no capítulo de resultados, suas iterações costumam ser muito intensas e geram muito trabalho. Portanto, poder fazer com que tal histórico de protótipos esteja organizado e fácil de consultar é primordial para tais equipes.

- **[RF 009] Permitir que os usuários possam executar as instâncias do game através do sistema:**

O sistema deve permitir que as instâncias armazenadas sejam executadas a partir dele, não criando necessidades extras de configuração e instalação de programas ou plug-ins adicionais para o usuário. A execução dessas instâncias deve ser tão simples e fácil quanto a execução de músicas em um *player*.

4.4.2 Requisitos Não Funcionais

- **[RNF 001] Permitir fácil acesso as informações presentes no sistema:**

Muitos dos entrevistados fizeram queixas ao tamanho dos GDDs e a quantidade de informações geradas, consideradas altas demais. Tamanho e quantidade acabam tornando difícil a navegação no GDD e muitas vezes, como atestaram os colaboradores da pesquisa e já vimos na literatura, afasta os desenvolvedores fazendo

com que estes não leiam o documento. Prover um meio de navegação simples em meio a tantas e diversas informações é essencial para diminuir os problemas mencionados.

- **[RNF 002] Impedir que o sistema seja intrusivo:**

As queixas ao tamanho dos GDDs, a sua quantidade de informações, entre outros problemas, faz com que a presença de tal artefato seja por muitas vezes intrusivo ao ambiente de produção dos desenvolvedores. Como já citado, muitos ignoram o documento e buscam a informação que precisam através de outros meios, normalmente a partir de uma conversa com o Game Designer. Portanto, o sistema deve impedir tais desconfortos para os desenvolvedores.

- **[RNF 003] O sistema deve permitir ao GDD simplicidade quando precisar ser modificado:**

Muitos dos entrevistados mencionaram que a manutenção do GDD é algo muito difícil de acontecer. Uma vez que o mesmo está pronto e a equipe produzindo, se torna difícil de parar uma atividade para atualizar o GDD [RNF 002] e quando ocorre a necessidade de consulta, o mesmo está desatualizado. Normalmente o GDD é longo, contém muito texto, demora para carregar e para se encontrar os pontos onde encaixar a modificação, fatores que afastam os usuários de realizar tal atividade e cria descompassos em relação ao design e ao desenvolvimento. De acordo com esse cenário, faz-se necessário que o GDD seja simples de ser atualizado, de fazer com que os usuários não precisem perder tanto tempo para acessar o documento e inserir as modificações que devem ser de domínio da equipe.

- **[RNF 004] O sistema deve permitir ao GDD estar sempre disponível:**

As mudanças durante o desenvolvimento de um game são muitas e a manutenção do GDD em meio tantas iterações e modificações têm sido um desafio para muitas equipes e diminui a confiabilidade no artefato, já que o mesmo não é atualizado no ritmo das modificações. Por isso, deve ser mantida uma estrutura de GDD fácil [RNF 003] para atualizar e que esta esteja sempre de prontidão para receber e distribuir a todos as modificações em seu conteúdo.

- **[RNF 005] Permitir que o sistema tenha fácil navegação no seu conteúdo:**

Como já visto em vários pontos deste trabalho, o GDD deve atender a públicos distintos, que usam formas e artefatos bem diversificados para se comunicar. O problema é que a grande quantidade de informações geradas pode (e deixa em muitos casos) deixar o GDD muito confuso e cheio de elementos desnecessários para uma parte ou outra da equipe. Por tal motivo o sistema precisa de um meio de organizar toda essa informação, para possibilitar que os diferentes públicos do GDD consigam navegar com facilidade no conteúdo do mesmo e encontrar mais rápido aquilo que necessitam.

5 Game Live Logs

A partir do conhecimento obtido com as considerações dos resultados da nossa pesquisa, decidimos transformar esse conhecimento em um protótipo para poder avaliar a adequação das nossas ideias (ou um subconjunto delas) através da experimentação em um contexto real de desenvolvimento de games. A vantagem é que além de ser um meio rápido de materializar o resultado da pesquisa, ganha-se uma ferramenta que permite encontrar inadequações, permitindo um maior aprendizado, a ser refletido nas gradativas evoluções da ideia [Calegario, F.C.A., 2013].

5.1 Influência

Decidimos tomar como principal referência, uma implementação do conceito de *Game Design Logs*, proposto por Daniel Cook e já explicado na seção 4.2. A nossa implementação, expande a proposta original corrigindo algumas de suas críticas e acrescentando os requisitos definidos de acordo com os princípios listados. Intitulamos o protótipo como **Game Live Logs** por inspiração de um dos colaboradores da pesquisa que mencionou a necessidade de a documentação de games ser encarada como um artefato “vivo”.

5.2 Running Lean e Metodologias Ágeis

Para implementação do protótipo, seguimos a abordagem *Running Lean* [Maruya, A. 2012] associada a práticas Ágeis [Wang, X. et al. 2012], mais especificamente o *Scrum*. A ideia é poder testar o mais rapidamente possível o sistema mais simples possível para evoluí-lo na direção certa, ao invés de construir um sistema completo ou complexo sem ter certeza de que ele atende às expectativas do cliente. Dessa forma, os requisitos listados anteriormente representaram nosso *Minimum Viable Product* (MVP).

Após a implementação iniciamos nosso ciclo de aprendizado, no qual avaliamos e atualizamos o nosso protótipo com novas versões, seguindo as recomendações de [Dow 2011] e [Thomke 2003] que indicam a criação de vários protótipos e a preparação para experimentações frequentes, rápidas e ágeis como fatores para se atingir soluções mais eficientes.

5.3 Protótipo versão 0 (V.0)

Considerando as características do sistema, criamos um protótipo que nos permitisse fazer uma avaliação rápida, mas que nos desse informações sobre como evoluir com o nosso sistema. O protótipo foi construído usando uma versão trial do Axure [Axure 2012], uma ferramenta muito popular para criação de protótipos para web. Nessa versão, o protótipo foi muito influenciado pela forma apresentada no *Game Design Logs* (ver Figura 5.1), com campos para inclusão de um tema e as soluções relacionadas, trazendo como novidade o campo para digitar a *tag* do log e a possibilidade de incluir anexos (ver Figura 5.2).

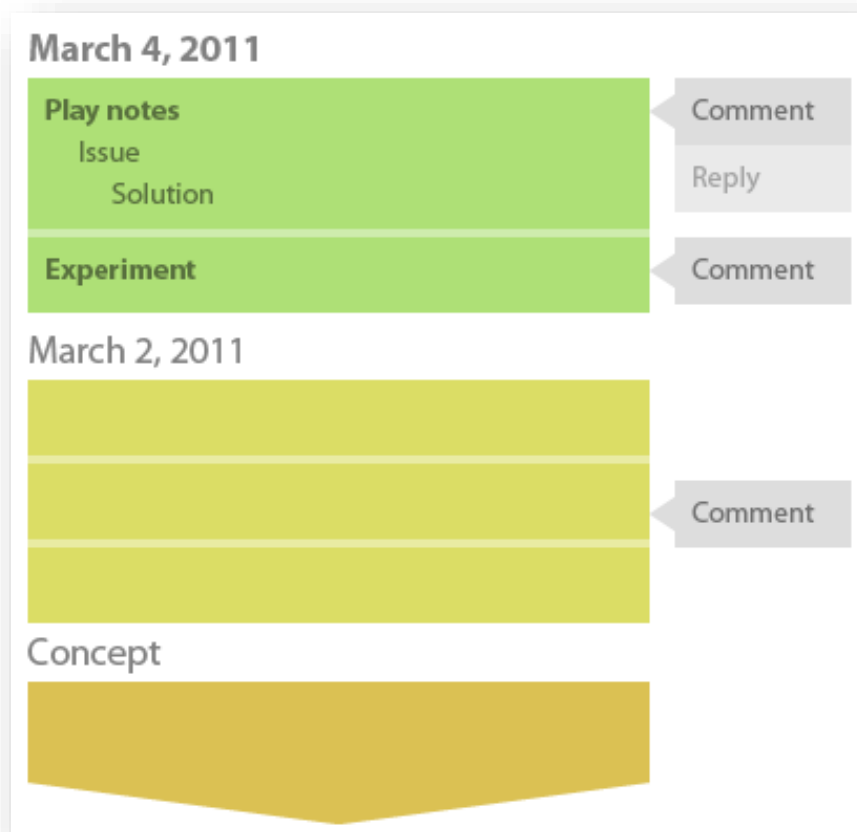


Figura 5.1: O formato do Game Design Logs [Cook, D 2011]

March 4, 2012		Tags	Design Programação
Play notes			
Issue	Solution		
Localização dos inimigos deixa jogador confuso sobre qual direção seguir na fase.	Mudar os inimigos para o trilho principal onde a ação da fase ocorre.		
Attachment			

Comments

acho que...

Tiago 05.03 14h03

acho que...

Tiago 05.03 15h03

acho que...

Tiago 05.03 16h03

Add Comment

Figura 5.2: Visualização de um log no nosso protótipo versão zero.

5.4 Primeira Avaliação

A avaliação do protótipo foi feita seguindo o método de avaliação “Rápida e Suja”, recomendado para coletar opiniões de usuários de forma informal acerca de elementos do sistema ainda em desenvolvimento [Sharp et al. 2007]. O protótipo foi apresentado a dois dos participantes das entrevistas, em momentos distintos. Inicialmente foi feita uma apresentação e após isso foi fornecido um tempo livre para que os usuários usassem o protótipo e fornecessem suas opiniões. Dentro da abordagem *Running Lean*, esses especialistas, até por conta da experiência no setor, representaram os nossos "usuários afoitos" (*early adopters*) e contribuíram com suas opiniões em vários momentos da evolução do protótipo.

5.5 Primeiros Resultados

Ambos os usuários mencionaram ter gostado da ideia dos logs, por tê-la considerado “fácil de seguir”, entretanto recomendaram ser importante não separar texto de conteúdo. Nessa versão do protótipo, todos os textos continham apontamentos para algum anexo associado que era aberto em nova aba (ver Figura 5.3).

Date	Issue	Solution	Author	Tags	Attachment
03.03	Localização dos	Mudar os inimi	Tiago	Design	--
01.03	Comandos de	Testar com	Tiago	Testes	--
26.02	Número de vidas	Estáticos	Tiago	Programação	--
24.02	Inimigos	Três tipos	Tiago	Arte; Progra	inimigo.png
22.02	Novos cenários	Não fazer	Tiago	Arte; Design	cena.png
20.02	Testes de sele	Acompanhar c/	Tiago	Programação	--
18.02	Prototipo em	Rodar no serv.	Tiago	Design; Pro	--
15.02	Versão 0.1	Cliente tem que	Tiago	Gerenciame	concept.doc
11.02	Concept sem s	Incluir seção de	Tiago	Design	--



Figura 5.3: A lista de logs e seus anexos associados. Os mesmos abriam em guias diferentes para visualização.

Os usuários revelaram que além de ser ruim para a escrita do GDD, tal separação possui problemas de usabilidade. A partir dessas opiniões identificamos problemas na navegação, porque os avaliadores precisavam ficar voltando nas páginas do sistema para inserir ou visualizar um log. Precisavam sempre visualizar uma lista de logs, para depois clicar em um específico e visualizá-lo, e caso o mesmo tivesse algum conteúdo anexado, precisariam clicar neste para que o mesmo fosse visualizado em outra página.

Com relação a inserção de *tags* e anexos, os campos respectivos, segundo os avaliadores, não estavam evidenciados e provocavam a omissão em algumas inserções. Além disso, os campos para digitar o texto dos logs foram considerados muito grandes e afastados demais um do outro, sem falar que os avaliadores consideraram que apenas um campo seria suficiente para indicar *issues* e *play notes*. De acordo com as opiniões dos usuários, modificamos as formas de inserir e visualizar os logs, com as novas propostas apresentadas nas (ver Figuras 5.4 e 5.5) na página a seguir.

Issue: campo para inserir a informação em forma de texto.

Tag: campo para inserir a categoria da informação, ou seja, se é uma informação mais relevante para a equipe de Design, de programação, de arte, etc.

Attachment: campo para incluir os anexos da informação

Submit: botão para enviar a informação para o sistema

Figura 5.4: protótipo da interface para inserir

Data de entrada da informação → 03.06.2013

Tag ou categoria → Tutorial

Autor → Julia

Texto do campo Issue → *Os tutorias do game devem ser disponibilizados em forma de vídeo, como na referência do game abaixo*

Exibição do conteúdo anexado →

Data de entrada do comentário → 03.06.2013

Autor do comentário → Tiago

Texto do comentário → *Gosto da idéia de usarmos vídeo, mas não seria mais interessante usar tutoriais jogáveis?*

Figura 5.5: protótipo da visualização dos logs.

A partir dos resultados iniciais, implementamos uma nova versão do protótipo usando os recursos do *Google App Engine*, o objetivo, além de obter um protótipo funcional e de alta fidelidade, foi o de apresentar nossos conceitos a indústria e angariar interessados que pudessem nos ajudar a avaliar e melhorar as futuras versões. Especificamente, utilizamos dois tipos de avaliação, ambos influenciados no processo *Running Lean*, o ciclo de aprendizado e a avaliação por especialistas (*Early Adopters*). Como nossa avaliação considerou testes com usuários, usamos o *framework* DECIDE.

5.6 Framework DECIDE

A escolha pelo DECIDE (acrônimo para Determine, Explore, Choose, Identify, Decide e Evaluate) considerou o fato de ser uma abordagem recomendada para avaliadores pouco experientes [Sharp et al. 2007].

O DECIDE, usa uma lista de decisões para guiar a avaliação, que são:

1. Determinar as metas da avaliação;
2. Explorar as questões a serem respondidas;
3. Escolher as técnicas da avaliação
4. Identificar questões práticas, como seleção dos participantes;
5. Decidir como lidar com questões éticas
6. Avaliar, interpretar e analisar os resultados.

Sendo assim, determinamos (1) como meta a melhoria da usabilidade do sistema de forma a atender os requisitos originados da pesquisa e a descoberta de novas funcionalidades que atendam às necessidades dos usuários que ainda não identificamos. A exploração (2) das questões foi feita por meio de entrevistas com os usuários. Para as técnicas (3) seguimos o protocolo explicado na próxima seção. As questões práticas (4), como participantes e ambiente de testes já foram abordadas na seção 5.7.1. Para as questões éticas (5), informamos que nenhuma dos usuários seriam revelados sem prévio consentimento. Nossa avaliação (6) foi realizada em ciclos (de forma iterativa), no qual a cada ciclo realizávamos entrevistas acerca da recepção dos usuários ao sistema e suas sugestões. Para a análise, a cada ciclo os usuários validavam os resultados das nossas

interpretações, frequentemente resultando em novas versões do sistema original atualizado com as melhorias requisitadas.

5.7 Ciclo de aprendizado

O ciclo de aprendizado teve a função de contribuir para a evolução do protótipo. Tal evolução ocorreu devido ao aprendizado obtido pelo contato com uma equipe profissional de desenvolvimento de games, que nos mostrou como melhorar nossa solução, tornando-a mais factível em relação aos problemas relatados.

5.7.1 Sujeitos

O protótipo foi testado pela empresa Manifesto [Manifesto Games 2012], uma empresa de games do Porto Digital do Recife que já possui diversos títulos lançados e cerca de oito anos de experiência. Mais precisamente, o teste foi realizado por uma das equipes de produção da empresa, durante um período de quatro semanas. Tal equipe foi selecionada por estar nos momentos iniciais do desenvolvimento de um game. A equipe era formada por um Game Designer, um Artista e um Programador.

5.7.2 Procedimento das avaliações

As avaliações foram feitas de forma iterativa. Inicialmente foi feita uma apresentação do protótipo e um vídeo tutorial foi disponibilizado para os participantes do teste. Os mesmos usaram a ferramenta livremente, mas a cada semana emitiam seus comentários sobre a mesma através de uma entrevista presencial com o pesquisador. O procedimento teve duas etapas, ambas apresentadas nas subseções abaixo.

5.7.2.1 Etapa 1 – Conhecimento e contextualização

Inicialmente o protótipo foi apresentado ao Produtor da empresa, bem como o contexto da pesquisa e o propósito da avaliação. Após a aceitação e confirmação da colaboração, chegou o momento de conhecer a equipe. A contextualização da pesquisa foi reapresentada, assim como algumas funcionalidades do protótipo. Esse primeiro momento foi executado rapidamente e durou algo em torno de trinta minutos. Ao final do

encontro, foi pedido que os membros da equipe fornecessem seus e-mails para cadastro na ferramenta.

5.7.2.2 Etapa 2 – Testes propriamente ditos

Assim que o protótipo foi disponibilizado para testes, a equipe recebeu um vídeo que apresentava como executar todas as funcionalidades disponíveis naquele momento.

Os usuários ficaram então livres para usar o protótipo quando e como achassem mais conveniente. Foi acordado também que receberiam visitas do pesquisador uma vez a cada semana.

5.7.2.3 Protocolo Cíclico de Avaliações

A cada semana o seguinte protocolo foi executado:

- 1 – Uma versão do protótipo era disponibilizada e o endereço para uso era divulgado aos usuários, bem como uma descrição das funcionalidades;
- 2 – Após uma semana da disponibilização, uma visita era feita.
 - 2.1 – Na visita os usuários apresentavam suas opiniões sobre o protótipo, incluindo queixas e sugestões de melhorias;
 - 2.2 – as queixas e sugestões eram anotadas e validadas pelos usuários.
- 3 – Uma nova versão, baseada nos resultados obtidos em (2) era disponibilizada;
- 4 – O ciclo de avaliações reiniciava.

5.8 Evolução do Protótipo

O protótipo disponibilizado tinha as características apresentadas no Quadro 5.1 logo abaixo:

Quadro 5.1: Características do Protótipo V.1
Funcionalidades:
<ul style="list-style-type: none">• Inserir Logs• Visualizar Logs• Consultar Logs<ul style="list-style-type: none">• Por Tag• Por Autor• Por Data
Arquivos suportados para exibição via browser
<ul style="list-style-type: none">• Vídeos de extensão: .mp4, .webm e .ogg• Áudio de extensão: .mp3• Imagens de extensão: .gif, .jpg, .png e .bmp
Browsers recomendados
<ul style="list-style-type: none">• Google Chrome Versão 25.0.1364.152m

Obs.: qualquer arquivo cuja visualização não fosse possibilitada pelo browser, tinha um link com seu conteúdo disponibilizado para download.

A interface do protótipo tinha as características apresentadas nas figuras abaixo, lembrando que na (ver Figura 5.6), os usuários fornecem os dados de entrada da classe *Log* (seção 7.5.1.1). Enquanto que na (ver Figura 5.7) os dados do *Log* são lidos do sistema para visualização. As consultas (ver Figura 5.8), permitem a recuperação rápida dos *logs*.

[Consultar por Tag](#) [Consultar por Autor](#) [Consultar por Data](#)
[Ver todos os Logs](#)

Game Logs

Insert a new Log:

Issue: Tag:

Log:

Upload Nenhum arquivo selecionado

Your latest Logs.

New Game Log

Issue: Incluir tema do Log.

Tag: Incluir a *Tag* do Log

Log: Incluir texto do Log

Upload: Upload de arquivos

Figuras 5.6 e 5.7: interface para inserir um *log*(*acima*) e interface de visualização de um *log* (*abaixo*).



Tags:

Flash Game SWF Apresentação [Mestrado]

Author:

tlam@cin.ufpe.br

Consultar por Data

Data Inicial:

Dia Mes Ano

Data Final:

Dia Mes Ano

Figura 5.8: as formas de consulta disponíveis

5.8.1 Primeira visita

Na primeira semana, os usuários não relataram nenhum problema acerca do uso do protótipo, tendo o considerado como “muito fácil de usar”. Além disso, não encontraram bugs ou falhas que comprometessem a atividade. Entretanto, consideraram que o protótipo precisa de muitos aperfeiçoamentos para chegar em um nível considerado como ideal. Na primeira versão do protótipo, o layout foi considerado o principal fator de incomodo, os usuários não gostaram de ser justificado à esquerda. O editor de texto, foi considerado muito limitado para o que os usuários relataram, posteriormente, ser preciso. Segundo os mesmos, um cuidado maior com a edição de texto era necessário e fizeram um esboço de como gostariam de editá-lo (Figura 5.9).

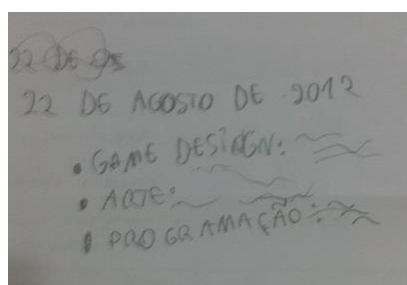


Figura 5.9: usuário explica através de um esboço como gostaria de editar o texto dos logs.

Uma listagem dos problemas encontrados e das melhorias sugeridas é apresentado na [Tabela 5.1] abaixo.

Tabela 5.1: problemas encontrados e melhorias sugeridas na primeira semana

Problemas encontrados	Melhorias sugeridas
Layout todo justificado à esquerda	Layout centralizado
Texto do Log possuía quantidade limitada de caracteres.	Texto sem limitação de caracteres
Texto sem parágrafo	Incluir parágrafos nos textos
Texto não permite marcadores	Incluir marcadores para os textos

5.8.2 Evolução do Protótipo – Primeira Semana

De acordo com o exposto anteriormente, fizemos atualizações no protótipo para contemplar as necessidades dos usuários. Mais especificamente, em relação ao layout, muitas mudanças aconteceram. A primeira versão usava apenas padrões HTML, o que criou a necessidade de incluir um arquivo CSS que permitisse editar mais facilmente detalhes da página como coloração de contêineres, tipo e tamanho de fontes, centralização entre outros. Quanto a limitação de caracteres no texto do Log, foi necessária uma mudança no tipo de dado que armazenava tal entrada. Foi deixado de usar o tipo **String** para usar o tipo **Text**, um tratamento especial da App Engine que permite a criação de texto sem limites de caracteres. Além disso precisamos incluir um editor que fornecesse mais opções do que a simples entrada de texto. Assim atualizamos a nossa interface com um editor de texto que permitisse a inclusão de marcadores esboçada pelo usuário e outras formatações como negrito e itálico por exemplo. Optamos por usar o **Widgeditor** [2012], um editor simplificado, de código aberto e de fácil integração com componentes HTML (*TextArea* nesse caso).

Considerando as modificações mencionadas acima, a segunda versão do protótipo apresentou as seguintes características (em vermelho estão as adições) no Quadro 5.2.

Quadro 5.2: Características do Protótipo V.2

Funcionalidades:

- Inserir Logs
 - Sem limites de caracteres
 - Com edição de texto
 - Negrito
 - Itálico
 - Hiperlink
 - Marcadores (ordenados e não ordenados)
 - Opção de cabeçalhos e parágrafos
- Visualizar Logs
- Consultar Logs
 - Por Tag
 - Por Autor
 - Por Data

Arquivos suportados para exibição via browser

- Vídeos de extensão: .mp4, .webm e .ogg
- Áudio de extensão: .mp3
- Imagens de extensão: .gif, .jpg, .png e .bmp

Browsers recomendados





- Google Chrome Versão 25.0.1364.152m

Por conta das novas funcionalidades a interface também apresentou mudanças no campo no qual se digita o texto do Log. Agora o mesmo possui um editor integrado que permite uso das características apresentadas na tabela de funcionalidades. Uma outra modificação foi feita na visualização dos Logs que, nessa segunda versão, apresentava todo o conteúdo de forma centralizada – os textos, os conteúdos associados e o comentários. Ambas as mudanças estão ilustradas nas (Figuras 5.10 e 5.11).

Insert a new Log:

Issue: Tag:

Log:

B **I**    

Texto do Log:

- Game Design
- Programação
- Arte

Listado com marcadores

Change block type
▼

- Heading 1
- Heading 2
- Heading 3
- Heading 4
- Heading 5
- Heading 6
- Paragraph

Upload Nenhum arquivo selecionado

Create Log

Figuras 5.10 e 5.11: a edição de texto (acima) recebeu um editor que fornece mais opções e o Log ganhou um layout (abaixo) que o centraliza, bem como o seu conteúdo.

Sat May 18 19:49:16 BRT 2013

Imagem do mario


Imagem

tulio

Imagem do Mario

- Colorida
- Media resolucao
- PNG

Anexo



your comments:

5.8.3 Segunda visita

Na segunda semana, o principal fato relatado pelos usuários foi a saída de um dos membros da equipe. Na ocasião houve uma substituição de programadores. Os próprios usuários trataram de apresentar o protótipo ao novo integrante, que concordou ser uma ferramenta fácil de utilizar e disse não ter precisado de maiores instruções para usá-la. Em relação as alterações, os usuários responderam de forma positiva tendo aprovado o novo layout e o as opções para o editor de texto. Como sugestão de melhoria, mencionaram um sistema de múltiplas *tags* no lugar de poder usar apenas uma. Segundo os mesmos, por mais que existam diferentes disciplinas é muito comum que um determinado assunto interesse a vários grupos distintos, como designers e programadores por exemplo, justificando assim a necessidade de múltiplas *tags*. A [Tabela 5.2] de problemas encontrados e melhorias sugeridas para a terceira versão do protótipo segue abaixo.

Tabela 5.2: problemas encontrados e melhorias sugeridas na segunda semana

Problemas encontrados	Melhorias sugeridas
Uso de uma única Tag	Sistema de múltiplas Tags

5.8.4 Evolução do Protótipo – Segunda Semana

A maior mudança em relação ao protótipo da versão anterior aconteceu devido as *tags* múltiplas. O sistema deixou de possuir um tipo de dado String que armazenava apenas uma *tag*, para possuir um *ArrayList*, que armazenava várias *tags* para um mesmo *Log*. Em relação a interface, nenhuma mudança drástica. O campo para inserção das *tags* continuou o mesmo, os usuários apenas receberam a instrução de separá-las por vírgulas quando necessitassem usar mais de uma.

Com a modificação das *tags*, a terceira versão do protótipo apresentou as seguintes características no Quadro 5.3.

Quadro 5.3: Características do Protótipo V.3

Funcionalidades:

- Inserir Logs
 - Sem limites de caracteres
 - Com edição de texto
 - Negrito
 - Itálico
 - Hiperlink
 - Marcadores (ordenados e não ordenados)
 - Opção de cabeçalhos e parágrafos
 - **Marcação por múltiplas tags**
- Visualizar Logs
- Consultar Logs
 - Por Tag
 - Por Autor
 - Por Data

Arquivos suportados para exibição via browser

- Vídeos de extensão: .mp4, .webm e .ogg
- Áudio de extensão: .mp3
- Imagens de extensão: .gif, .jpg, .png e .bmp

Browsers recomendados

- Google Chrome Versão 25.0.1364.152m

5.8.5 Terceira visita

Na terceira semana, os usuários sentiram a necessidade de visualizar conteúdos produzidos na ferramenta **Adobe Flash**. Muito da produção da equipe envolve a criação de conteúdo em tal ferramenta, e por isso mesmo, revelaram que uma maneira de visualizar mais rápido e fácil tais produções seria uma grande contribuição ao trabalho do time. Um outro pedido foi o uso de *tags* pré-definidas. Os usuários aprovaram a implementação de múltiplas *tags*, mas sugeriram que pelo menos as mais usadas (que no caso da equipe são: Design, Artes e Programação) fossem pré-definidas para evitar ter sempre que digitá-las novamente. De acordo com os pedidos apresentados, a [Tabela 5.3] de problemas encontrados x melhorias sugeridas apresentou a configuração abaixo:

Tabela 5.3: problemas encontrados e melhorias sugeridas na terceira semana

Problemas encontrados	Melhorias sugeridas
Necessidade de visualizar conteúdo produzido com Flash.	Extensão da visualização dos Logs para visualizar conteúdo com extensão .swf .
Necessidade de usar <i>tags</i> pré-definidas	Criar um menu com as <i>tags</i> mais usadas no campo de inserção de <i>tags</i> .

5.8.6 Evolução do Protótipo – Terceira semana

A mudança mais significativa em relação a versão anterior foi a visualização de conteúdos produzidos no Flash. Com isso, qualquer arquivo de extensão **.swf** podia ser facilmente armazenado e visualizado. A outra mudança foi a implementação de um menu com as *tags* indicadas pelos participantes da pesquisa (**Figura 5.12**).

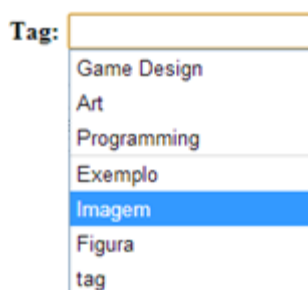


Figura 5.12: tags pré definidas

Essa nova característica ainda possui a vantagem de usar o armazenamento do browser, que preserva as entradas do usuário para que não seja preciso redigitá-las no futuro. Assim, os usuários podem optar se vão criar novas *tags* ou se as preexistentes são suficientes para categorização do Log. De acordo com as novas implementações o protótipo passou a apresentar as seguintes características no Quadro 5.4:

Quadro 5.4: Características do Protótipo V.4

Funcionalidades:

- Inserir Logs
 - Sem limites de caracteres
 - Com edição de texto
 - Negrito
 - Itálico
 - Hiperlink
 - Marcadores (ordenados e não ordenados)
 - Opção de cabeçalhos e parágrafos
 - Marcação por múltiplas *tags*
 - **Uso de *tags* pré-definidas**
- Visualizar Logs
- Consultar Logs
 - Por Tag
 - Por Autor
 - Por Data

Arquivos suportados para exibição via browser

- Vídeos de extensão: .mp4, .webm e .ogg
- Áudio de extensão: .mp3
- Imagens de extensão: .gif, .jpg, .png e .bmp
- **Exibição de conteúdos produzidos em Flash: .swf**

Browsers recomendados

- Google Chrome Versão 25.0.1364.152m

5.8.7 Quarta visita

Na quarta semana, os usuários mencionaram o uso de um software de gerenciamento de projetos, o Jira [2012]. Por meio desse software são listadas e gerenciadas as tarefas da equipe. Segundo os usuários, seria interessante uma comunicação entre o protótipo e o Jira, pois muito do que acontecia em um dos ambientes acabava exercendo influência no outro. Essa revelação foi interessante, pois é diretamente relacionada a fazer com que a conversação dos Logs possa ser transformada em dados que auxilie a gerência do projeto. Os usuários recomendaram a criação de um campo para que eles pudessem inserir o número da tarefa no Jira ao qual o(s) Log(s) tinham relação. Um outro pedido dos usuários foi uma forma de se fazer marcação diferenciada para textos contendo blocos de código. Segundo os mesmos, as vezes algumas definições são passadas como uma lista em *tags* de XML e a formatação comum do protótipo não é a mais adequada para tal visualização. Além disso, mencionaram o fato de facilitar a conversação entre os programadores. Por fim, foi pedido para que todos os *Logs* pudessem ser visualizados na página principal (em todas as versões só os cinco *Logs* mais recentes eram vistos na página principal. Todos os Logs poderiam ser conferidos clicando em um link para tal função). Segundo os usuários, essa é a melhor forma de visualizar a proposta de documentação alternativa, sugerida inicialmente pelo criador do Game Design Logs, Daniel Cook. Ainda segundo os usuários, isso facilita a visualização da progressão dos *Logs*, mantém a noção de histórico e diminui o tempo de espera em carregar uma outra página para essa função. Desse modo, a criação de novos *Logs* e os *Logs* já preservados ocupam sempre o mesmo espaço. Com relação as atualizações, os usuários aprovaram a forma de uso das *tags* pré-definidas e foram enfáticos acerca da visualização de conteúdos produzidos com Flash. ***“Você leu nossos pensamentos!”*** foi a resposta obtida quando perguntado se tal funcionalidade estava pronta do jeito que eles desejavam. A seguir a [Tabela 5.4] de problemas encontrados x melhorias sugeridas de acordo com os comentários dos usuários.

Tabela 5.4: problemas encontrados e melhorias sugeridas na quarta semana

Problemas encontrados	Melhorias sugeridas
Necessidade de associar Logs com tarefas na ferramenta Jira	Inclusão de um campo nos Logs para indicar o número da tarefa no Jira.
Necessidade de formatação (e highlight) de código	Incluir opção para formatar texto como código no editor
Necessidade de visualizar todos os Logs na mesma página de criação	Exibir todos os Logs e não apenas os mais recentes na página de criação

5.8.8 Evolução do Protótipo – Quarta semana

Nessa quarta semana, mais até do que nas anteriores, as mudanças sugeridas não surtiram em grande esforço de implementação, particularmente a inclusão de um campo a mais para o registro da tarefa ao qual Log era associado foi uma alteração extremamente simples de implementar e o resultado na interface com o usuário pode ser conferido na figura abaixo (ver Figura 5.13).

Insert a new Log:

Issue: Tag: Task Number:

Log:

Rich text editor toolbar: B, I, link, list, code, Change block type

Figura 5.13: Formulário de criação de Logs, agora modificado com o campo para indicar tarefa associada em ferramenta de gerenciamento de projetos.

Para a formatação e marcação de código incluímos uma opção a mais na lista de escolhas de formatos para blocos de texto do **Widgeditor** (Figura 5.14). Quando selecionada a mesma ativava o script **highlight.js** [2012], uma solução popular e gratuita

para esse tipo de serviço, e o mesmo se encarregava de adequar o bloco de texto selecionado, colorindo palavras reservadas e outros símbolos de linguagens de programação.

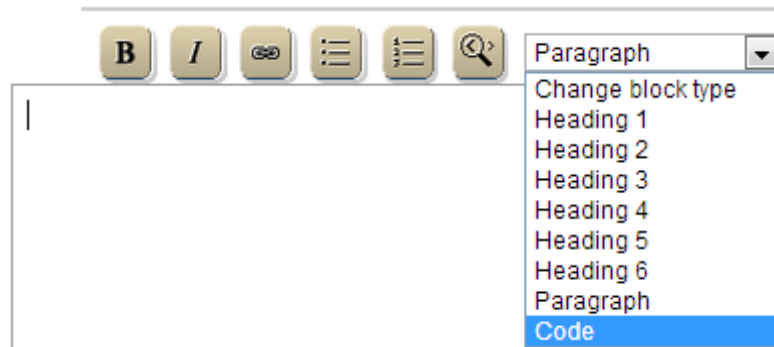


Figura 5.14: Para formatar um bloco de texto como código, os usuários bastam apenas selecioná-lo e escolher a opção “Code” (no destaque em azul), exatamente como as outras opções de formatação.

Em relação a visualização dos *Logs* na página principal (que é a página de criação), bastou modificar as linhas de código que continham a instrução para exibir os cinco *Logs* mais recentes e assim permitir que todos os *Logs* pudessem ser exibidos, conforme a vontade dos usuários.

5.9 Comparativo de ferramentas

Durante as várias semanas de teste, perguntamos aos usuários como eles fariam para manter os *Logs* caso não estivessem usando nossa ferramenta. Eles nos responderam que tentaram, sem sucesso, manter tal prática em outra oportunidade usando o Facebook. Os problemas em usar tal abordagem, segundo os usuários, é que ao decorrer do tempo fica difícil de achar postagens antigas, a organização não é adequada para um sistema de registros, há muita interrupção dos círculos sociais e etc. Após a experiência com o Facebook, os usuários relataram que passaram a usar o phpBB [2012]. Trata-se de um sistema de fórum de código aberto e gratuito, usado para manter discussão acerca de vários projetos mantidos pela Manifesto.

No caso da equipe de testes, em seus projetos, cada informação considerada de importância para equipe era postada no fórum. Os problemas residem no fato da organização de tópicos de um fórum ser estruturada de uma forma diferente da necessária pelo sistema de *Logs*. Além disso, consultando o próprio fórum do phpBB vemos que o

sistema, até pela sua característica de ser de código aberto, possui várias extensões para dar suporte a conteúdos específicos (como postagem de arquivos .swf) e um esquema de *tags*. Entretanto, muitos desses não são bem classificados e sofrem muitas críticas de seus usuários. Fizemos um apanhado das principais características para um sistema de Logs como o pretendido pela nossa aplicação e os comparamos com as soluções apresentadas tanto pelo nosso protótipo, bem como pelas ferramentas citadas (*Facebook* e *phpBB*). Baseado nas opiniões dos usuários fizemos uma tabela [Tabela 5.5] com as impressões sobre o uso de cada uma das soluções apresentada a seguir.

Tabela 5.5: As ferramentas e impressões sobre as características apresentadas.

	Facebook	phpBB	Protótipo
Editor	Muito simplificado	Completo	Possui as necessidades básicas.
Tags	Insuficiente para os propósitos dos Logs.	Não possui*	Diretamente relacionadas aos Logs.
Conteúdo	Suporte a maioria dos conteúdos necessários, mas nada específico	Suporte a maioria dos conteúdos necessários. Extensões para conteúdos específicos apresentam problemas.	Suporte a maioria dos conteúdos necessários e funciona com conteúdos específicos sem problemas.
Organização	Pouca estrutura para organização, principalmente a medida que os posts aumentam	Bem estruturado, porém o modelo da estrutura de fórum é insuficiente para os propósitos dos Logs.	Possui a estrutura necessária para a organização dos Logs.
Buscas	Não possui	Sistema de buscas em fórum. Pouco preciso para os Logs.	Sistema facilitado pelas tags.
Chat	Completo	Extensão que apresenta problemas	Não possui
Alertas	Funciona através de notificações no site e por email	Extensão que funciona enviando alertas por email	Não possui

* : foram encontradas implementações de sistemas de *tags* no fórum do phpBB, mas todas analisadas apresentam muitas críticas dos usuários.

De acordo com tais opiniões consideramos que os próximos passos para evolução do nosso sistema é a implementação de um serviço de *chat* e de *alertas*. A API da App Engine oferece recursos para tais serviços, o que facilita a inclusão dessas características no nosso sistema sem o perigo das inconsistências que ocorrem com as extensões do phpBB e com a vantagem de poder integra-los a outros serviços Google como o Google Calendar.

5.10 Avaliação dos usuários afoitos

Na metodologia *Running Lean*, o conceito de *Usuários Afoitos* remete a pessoas interessadas e motivadas pelo produto ou serviço em desenvolvimento que podem facilmente influenciar outras pessoas a usá-lo. No nosso trabalho, tivemos contato com vários especialistas, profissionais com um bom entendimento do problema em questão e com bastante experiência na indústria. Identificamos três desses profissionais como *Usuários Afoitos*, pela coloboração em vários momentos da pesquisa e sobretudo pelo interesse em utilizar os resultados em suas atividades. Para avaliar a opinião dos *Usuários Afoitos*, usamos a entrevista com especialistas. O uso de tal avaliação tem sido cada vez mais comum na indústria de software [Mustapic, G. et al 2004] e se revelado como um meio valioso de obter informações importantes [Garcia, Vinicius C. 2011]. A técnica apresenta também uma diversidade de métodos em sua aplicação como a análise qualitativa de conteúdo presente em [Magenheim, J. et al 2010] e adotada para este trabalho. Usamos as recomendações de [Meuser, M. and Nagel, U 2002] e a direcionamos em forma semi-estruturada de acordo com o roteiro presente em [ANEXO 2] e usando o mesmo modo de análise como apresentada no capítulo 3. Todas as entrevistas foram realizadas ao final das evoluções do protótipo, sendo duas presenciais e uma a distância através de e-mail. Basicamente os *Usuários Afoitos* responderam acerca de melhorias e cenários nos quais o sistema apresenta soluções interessantes. Abaixo apresentamos algumas das citações, separadas em categorias proveniente da análise de dados, dos nossos participantes.

5.10.1 Facilidade no Uso

“Achei fácil de usar, sim. Mas a usabilidade e a UI da ferramenta precisam ser modernizados para poder viabilizá-la no mercado. E eu acho que existe uma oportunidade para ela.”

De forma geral, os entrevistados não relataram problemas no uso da ferramenta. Mas recomendaram mais investimento em UX (*User Experience*), para que o protótipo ganhe mais em usabilidade.

5.10.2 Melhorias sugeridas

“Uma funcionalidade que pensei agora é a de se gerar uma versão para impressão. Isto seria interessante para o time anexar uma documentação junto com as snapshots/releases dos produtos. Por mais que as informações possam (e vão) mudar com o tempo, é importante registrar o estado em que o GDL estava na época de algum milestone.”

“Acredito que uma boa melhoria seria um serviço de notificação que indicasse a entrada de novos dados no sistema” “Seria interessante você mudar alguns nomes de campos e botões, como você tem uma preocupação em atender diversos públicos é importante que você esteja preparado para lidar com pessoas que têm um vocabulário de mundo profissional bem diferente do nosso. Enfrentamos tal situação nos nossos projetos, temos muitos consultores da área de educação e acredito que pra eles termos como ‘issue’ e ‘log’ possam não ser tão esclarecedores quanto é pra nós”

O interessante das citações acima é que dois dos princípios foram abordados no discurso dos usuários. A versão de impressão, sem dúvida é uma melhoria interessante e diz respeito diretamente ao princípio de forma e conteúdo, pode ser inclusive uma solução para o problema citado na seção de críticas do capítulo 2 sobre a dificuldade de imprimir GDDs editados em páginas **Wiki**. Já a preocupação com os termos técnicos da interface, que, segundo o entrevistado, podem não ter o mesmo apelo com determinados profissionais é diretamente relacionado ao princípio de público – alvo.

5.10.3 Cenários

“Essa característica de poder visualizar conteúdo em Flash é uma grande ajuda. No meu caso, eu tenho que avaliar muito desse conteúdo e acabo muitas vezes abrindo um monte de players, fico com a tela do desktop cheia. Com essa ferramenta, que deixa tudo no mesmo lugar, resolve esse problema”

Um dos entrevistados, conseguiu ser bastante específico, citando a visualização de conteúdo produzido em Flash. Em seu caso, devido ao grande número de avaliações que ele precisa fazer o protótipo funciona como aliado mantendo todos os arquivos em um mesmo local e funcionando sem ter que executar vários players ou abrir vários players.

5.11 Conclusões da Avaliação

Nas quatro semanas de teste, observamos que a evolução do protótipo seguiu os princípios que tínhamos previamente definidos. Ao longo dos ciclos de avaliação, aos poucos, os usuários moldaram várias das características planejadas, mas que ainda não tinham sido concretizadas, exatamente por uma necessidade de uma experiência prática. Mais precisamente, essas características foram quanto a necessidade de um editor mais elaborado, visualização de conteúdo específico e a relação com a gerência de projetos.

Tecnicamente falando, a escolha do Google App Engine se mostrou perfeitamente adequada ao tipo de projeto. Nenhum problema foi relatado pelos usuários acerca de lentidão, de falta de capacidade para armazenamento, de sobrecarga, de páginas não encontradas, etc. Além disso, o próprio ambiente do App Engine oferece uma série de recursos que permite verificar o uso da aplicação, fornecendo inclusive relatórios de uso dos usuários, armazenamento, visualização dos dados e etc (**Figura 5.15**).

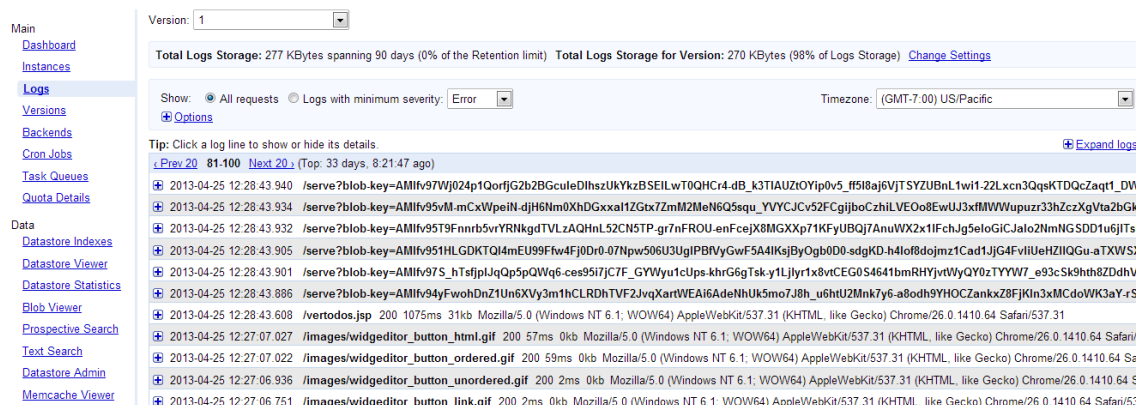


Figura 5.15: Uma das inúmeras telas do painel de controle da aplicação fornecido pelo App Engine. Na imagem, vemos uma série de registros que indicam a criação de um novo Log.

Em relação aos *Early Adopters*, suas contribuições revelaram a necessidade de um maior investimento em UX para melhorar o uso do sistema e nos mostrou que os princípios envolvidos na implementação do protótipo são coerentes com o cotidiano das suas atividades. Por fim, forneceram opiniões entusiasmadas sobre o protótipo avaliado, encorajando a continuidade do projeto e enxergado novos rumos para o mesmo, como na citação abaixo.

“Eu teria total interesse em usar a ferramenta! Principalmente numa plataforma de SaaS (Software as a Service). O conceito está muito bom, e acredito que ela tem um bom espaço para crescer no mercado.”

A seguir compartilhamos nossas considerações sobre como o protótipo do **GameLiveLog** foi se aproximando dos princípios do projeto, bem como nossos apontamentos acerca das direções nas quais o mesmo deve evoluir.

5.11.1 Público – Alvo diversificado

Durante as semanas de teste, o protótipo iniciou com três usuários, ocorreram mudanças na equipe e no decorrer das semanas outros dois interessados pediram para ter acesso ao sistema. Considerando todos que usaram o sistema por pelo menos uma semana, tivemos ao todo seis usuários: um produtor, um diretor de arte, um game designer, um artista e dois programadores.

Não houve de nenhum deles queixa quanto a ter que ler informação desnecessária ou algo do tipo e segundo os próprios usuários a facilidade de uso da ferramenta facilitava a consulta e o acompanhamento dos logs. Além disso, várias modificações, como um novo sistema de *tags* e a formatação de blocos de texto como código foram melhorias que aos poucos adaptaram o protótipo a públicos distintos.

5.11.2 Diversidade de formas e conteúdos

Uma grande queixa aos GDDs é exatamente a falta de suporte que os mesmos fornecem a diversos tipos de conteúdo. Por isso, quando permitimos o uso de conteúdo produzido em Flash nos Logs, os usuários disseram que tínhamos “*lido seus pensamentos*”. Há inúmeros tipos de conteúdo que as equipes de desenvolvimento de games utilizam, prover um meio de visualizar facilmente esse conteúdo, como fizéssemos nesse caso particular, é de fundamental importância para esses times e obviamente deve continuar a ser um dos principais objetivos deste projeto. Além disso, deve-se investir em formas de visualização que atendam adequadamente a essa diversidade de conteúdo.

5.11.3 Imposição do GDD por culturas corporativas

Durante os testes, em nenhum momento os usuários se queixaram de o protótipo não permitir que eles pudessem fazer uso das práticas normalmente adotadas por sua corporação. Perguntamos se o protótipo estava sendo intrusivo nesse sentido ou se estava prejudicando em algum ponto as atividades cotidianas. Os usuários responderam que não sentiram nenhum desconforto nesse ponto ocasionado pelo uso do protótipo e ainda afirmaram que a facilidade em usá-lo impediu situações desse tipo.

5.11.4 Evolução do GDD

Como dito anteriormente, o GDD é um artefato dinâmico e que nunca está finalizado. Por conta dessas características o nosso protótipo deveria ser propício a esse dinamismo. Durante o período de testes, tivemos as mudanças relacionadas ao projeto, bem como mudanças dentro da equipe. Nesse cenário, não registramos nenhuma queixa dos usuários mencionando que o protótipo não dava suporte as evoluções do projeto, muito pelo contrário, os usuários afirmaram várias vezes que o grande benefício do

protótipo é sua facilidade de uso, principalmente no que consiste a postar novos Logs sempre que necessário.

5.11.5 Documentação e o plano de projeto

Uma das preocupações era que o sistema permitisse a saída de um ambiente de discussão de design para um ambiente de produção. Em outras palavras, como apontado em vários momentos da pesquisa é importante que a partir do design do game, a produção possa usar essas informações para definições de tarefas e outras atividades relacionadas ao gerenciamento. Durante os testes, observamos isso ocorrer quando os usuários pediram que criássemos um campo no Log que pudesse vincular o mesmo a uma tarefa criada no sistema de gerenciamento Jira. Mais ainda, os usuários sugeriram buscas baseadas nesse vínculo. Acreditamos que a melhor evolução nesse sentido é a criação de um *plugin* que permita aos usuários a rápida mudança entre a visão de design (obtida com os Logs) para uma visão de projeto (Jira ou sistemas equivalentes).

5.11.6 Código como documento

No momento o protótipo pode fornecer o download de qualquer executável do game em desenvolvimento, entretanto, embora desse modo já seja possível manter todas as versões do game em um mesmo ambiente, acreditamos que o melhor é dispor de um meio de rodar tais versões dentro do sistema. Tanto os usuários que avaliaram o GLL em no trabalho cotidiano, quanto os *Usuários Afoitos* mostraram entusiasmo em poder executar conteúdo Flash, sem ter que usar players extras e recursos do tipo. Nas próximas evoluções do protótipo, deve haver investimento em dispor de um meio semelhante que execute outros formatos (como .exe) para atender melhor esse princípio.

Por fim apresentamos na página seguinte como ficou o quadro de características do protótipo após as quatro semanas nas quais o GLL foi avaliado. Na próxima seção, apresentaremos alguns detalhes técnicos da implementação do GLL e em seguida nosso capítulo de contribuições e trabalhos futuros.

Características do Protótipo V.5

Funcionalidades:

- Inserir Logs
 - Sem limites de caracteres
 - Com edição de texto
 - Negrito
 - Itálico
 - Hiperlink
 - Marcadores (ordenados e não ordenados)
 - Opção de cabeçalhos e parágrafos
 - Marcação por múltiplas tags
 - Uso de tags pré-definidas
 - Campo de número da tarefa (associação com ferramenta de gerenciamento de projetos)
 - Formatação para códigos (Programação)
- Visualizar Logs
 - Visualização de todos os Logs na página principal
- Consultar Logs
 - Por Tag
 - Por Autor
 - Por Data

Arquivos suportados para exibição via browser

- Vídeos de extensão: .mp4, .webm e .ogg
- Áudio de extensão: .mp3
- Imagens de extensão: .gif, .jpg, .png e .bmp
- Exibição de conteúdos produzidos em Flash: .swf

Browsers recomendados

- Google Chrome Versão 25.0.1364.152m

6 Implementação: Definições técnicas

Como apresentado na seção 6.2, uma das necessidades mais importantes do nosso sistema é a sua disponibilidade. O cenário de desenvolvimento de games é intenso e dinâmico e mudanças ocorrem o tempo todo, por isso é importante que qualquer sistema que tenha intenção de auxiliar tal desenvolvimento possa estar sempre à disposição das equipes.

6.1 Game Design nas nuvens

Considerando esse fato, decidimos que o Game Live Logs deveria ser publicado na *web*, como uma aplicação que está sempre rodando e de prontidão para as ações de seus usuários. Por conta disso consideramos usar algum sistema de Computação nas Nuvens [Buyya, R. et al 2009] que desse suporte a nossa aplicação. Os principais analisados foram, Amazon AWS, [2012] Google App Engine [2012] e Windows Azure [2012]. Após consulta aos sites oficiais das ferramentas, vimos que a escolha não é uma tarefa trivial, principalmente quando se pretende fazer algum investimento financeiro em uma das opções disponíveis, como atesta as pesquisas de [Li, A. et al. 2010]. No nosso caso, era importante contar com uma alternativa considerada fácil para iniciantes em aplicações na Nuvem, que tivesse configuração e manutenção simples (de preferência em alguma *Integrated Development Environment* - IDE conhecida), que possibilitasse teste gratuito sem limite de tempo e que servisse como um ponto de partida para provar conceitos. Dentro dessas considerações, escolhemos o Google App Engine.

6.2 Google App Engine e a estrutura do projeto

A estrutura do projeto teve inspiração no modelo Model-View-Control (MVC), tal escolha foi tomada por se tratar de uma aplicação Cliente – Servidor cujo modelo tem sido referenciado positivamente em uma série de trabalhos, servindo como base para implementação de novos Frameworks como em [Stäbler, A. 2008], [Fernandez, L. et al. 2009] e [Kuuskeri, J. 2011]. A IDE escolhida foi o **Eclipse** [2012], nas configurações indicadas pelo Google em sua página de primeiros passos da App Engine.

Por fim, o sistema como um todo foi dividido em *Backend* e *Frontend*, que representam respectivamente o lado Servidor e Cliente da aplicação.

6.2.1 Backend

O *Backend* foi todo implementado em Java, em uma estrutura semelhante a do MVC. Essa parte do sistema representou as camadas *Models* e *Controls*. Criamos uma camada extra, denominada *Managers*, cujo objetivo é a criação da utilização das interfaces que cuidam dos serviços de persistência da aplicação.

6.2.1.1 Camada Models

Na camada *Models*, incluímos as classes básicas do sistema: *Log* e *Comment*. A classe *Log* é a responsável por toda a edição de conteúdo do sistema, seja ele um simples texto, um vídeo ou um conteúdo interativo e inclusive, todas essas opções juntas. Já a classe *Comment*, como o próprio nome sugere serve para comentar os logs.

- **Classe Log**

É o coração do sistema. Toda conversação tem base com a inclusão de um *log*, este pode ser uma simples mensagem ou um texto contendo qualquer tipo de anexo que dê suporte a informação a ser transmitida. Como a classe tem associação com a maioria dos requisitos do sistema, mais precisamente os [RF 001], [RF 002], [RF 003], [RF 007], [RF 008] e [RNF 005], faremos a seguir uma pequena descrição dos atributos presentes:

Key – Atributo key do tipo Key, este é a chave principal de cada *Log*, todos os *logs* terão chaves diferenciadas e por isso mesmo serão únicos, fáceis de registrar e recuperar no sistema.

Author – Atributo do tipo User que guarda informações acerca do autor do *Log*, como e-mail por exemplo.

Tag – Atributo do tipo String que permite criar categorias para os *logs*, possibilitando que os usuários encontrem mais fácil o que buscam e direcionem a informação também para públicos específicos.

Issue – Atributo do tipo String que representa o tema a ser tratado no *log*.

Content – Atributo do tipo String que contém o texto do *log*.

Date – Atributo do tipo *Date* que armazena a data de criação do log.

BlobKey – Atributo do tipo *BlobKey*, usado para registro da chave de acesso aos anexos dos logs.

- **Classe Comment**

A classe *Comment* é a responsável por manter a conversação dos logs ativa, caso os usuários necessitem. Seu uso remete as dúvidas e sugestões relacionados com o *log* do qual se associa.

6.2.1.2 Camada Managers

A camada Managers implementa as interfaces *EMFService* e *PMFService*.

- **EMFService**

O objetivo de tal classe é usar uma interface *EntityManagerFactory*, cuja finalidade é fornecer serviços para manipulação de dados persistentes.

- **PMFService**

O objetivo de tal classe é usar um objeto do tipo *PersistenceManagerFactory*, cuja finalidade neste trabalho é fornecer serviços para recuperação de dados persistentes. A *EntityManagerFactory* faz parte do pacote *javax.persistence* e a *PersistenceManagerFactory* é do pacote *javax.jdo.PersistenceManagerFactory*. Ambos os pacotes fazem parte da distribuição original da App Engine, não sendo necessárias configurações extras no projeto para usar essas classes.

6.2.1.3 Camada Controls

A camada de controle cuida das ações do usuário, como inserção e remoção de logs por exemplo. É constituído das classes, *InsertLog*, *RemoveLog*, *ServeLog* e *InsertComment*.

- **InsertLog**

Classe responsável por montar a estrutura dos *Logs* a partir de dados fornecidos pelas entradas do usuário. Tais dados são armazenados em atributos do tipo *Log* e passam a ser persistentes.

- **RemoveLog**

Como o próprio nome já indica, a função de tal classe é a remoção de um determinado *log*. A remoção de um *log*, remove automaticamente todos os comentários associados.

- **ReceiveLog**

Classe responsável por receber e tornar disponíveis os conteúdos (vídeos, imagens, sons, planilhas, documentos de texto, etc.) associados aos logs.

- **InsertComment**

Classe responsável por criar os comentários e os associar com os logs. Todos os comentários de um mesmo log devem possuir como chave externa, a chave principal do objeto log aos quais estão associados.

6.2.2 Frontend

Para a implementação do *Frontend* da aplicação usamos Java Server Pages (JSP), HTML5, Javascript e Cascade Style Sheet (CSS). O JSP combina códigos Java com HTML 5 e o utilizamos devido aos muitos exemplos fornecidos nas páginas oficiais da documentação da App Engine que tornaram seu aprendizado fácil, rápido e suficiente para os propósitos desta pesquisa. O *Frontend* da aplicação fica concentrado no pacote *Web Application Resources (WAR)* e internamente contém os seguintes diretórios:

- **CSS**

Armazena os arquivos CSS responsáveis pelo layout da aplicação.

- **Images**

Armazena os arquivos de imagens estáticos da aplicação, como backgrounds, ícones e etc.

- **Scripts**

Armazena os arquivos em Javascript que cuidam de diversos aspectos da interação como o lançamento de alertas para o usuário e formatação de texto.

- **WEB-INF**

Este é um diretório que contém arquivos e outros diretórios relacionados a configuração do projeto, como o mapeamento dos *servlets* apresentados anteriormente.

- **Páginas HTML e JSP**

Estas páginas são as responsáveis por renderizar a interface com o usuário e possibilitar ao mesmo o uso dos serviços da aplicação.

6.2.3 Comunicação Backend – Frontend

Toda a comunicação entre os lados *Frontend* e *Backend* é realizada através de *Servlets*, um componente como um servidor, que gera dados para a camada de apresentação da aplicação. Seu funcionamento é quase como o de uma classe Java que dinamicamente processa requisições e respostas.

A configuração dos Servlets é feita no arquivo web.xml, chamado de descritor da aplicação. Está contido no diretório WEB-INF. Essa configuração mapeia as classes Java que realizam os serviços requisitados pelos usuários interagindo no lado Frontend da aplicação. Assim que a solicitação é recebida no lado Backend, o descritor mapeia a classe que realizará o serviço e após conclusão do mesmo, a resposta à requisição é enviada para o usuário.

Tais requisições são realizadas mediante métodos do protocolo HTTP como GET e POST. Na nossa aplicação, as classes da camada Controls herdam as propriedades da classe HttpServlet, o que facilita todo o processo de controlar as requisições HTTP e a comunicação com o descritor de conteúdo.

6.3 Visualização e representação da interação do usuário com o sistema

Nesta seção, apresentaremos uma visualização geral da arquitetura do sistema com os diagramas das classes principais e uma representação de como o sistema se comporta a partir da interação do usuário.

6.3.1 Pacote Models

O pacote *models* (ver Figura 6.1) contém as classes que representam os principais dados da aplicação, a classe *Log* e a classe *Comment*. Entre os atributos, merecem destaque o tipo *Key* – usando para tornar todo novo *Log* (ou *Comment*) persistente na aplicação e o tipo *BlobKey*, usado para armazenar persistentemente arquivos multimídia. Com o atributo *Key* podemos associar dados, dessa forma podemos criar associações de vários comentários para um mesmo *Log*.

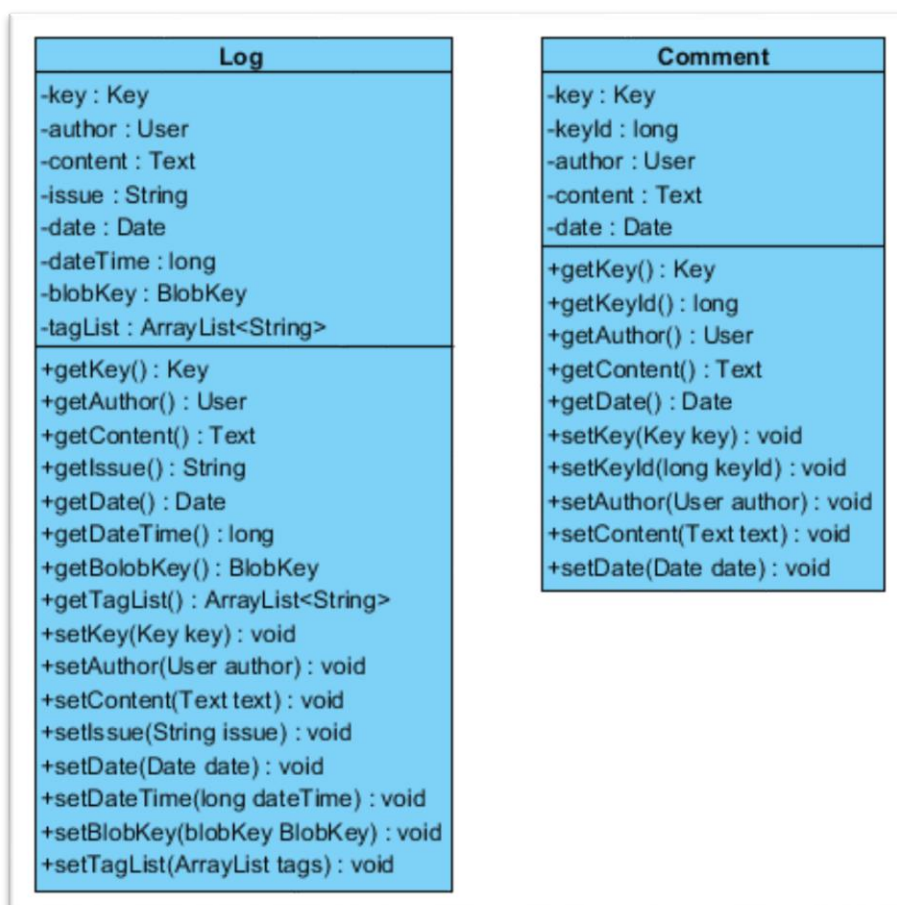


Figura 6.1: Classes do pacote *models*.

6.3.2 Pacote Controls

Este pacote contém as classes que implementam os serviços decorrentes da ação dos usuários (ver Figura 6.2). Por meio de páginas HTML, os usuários passam seus parâmetros para criação, atualização, busca e remoção de *Logs* e *Comments*. A partir de *Servlets*, as requisições mapeiam as classes que implementam o serviço requisitado pelo usuário e atualizam as páginas HTML com as respostas. Nesse pacote, todas as classes herdam de *HttpServlet* que já contém implementações que encapsulam os métodos GET, POST, entre outros necessários para aplicações cliente e servidor que usam o protocolo HTTP.

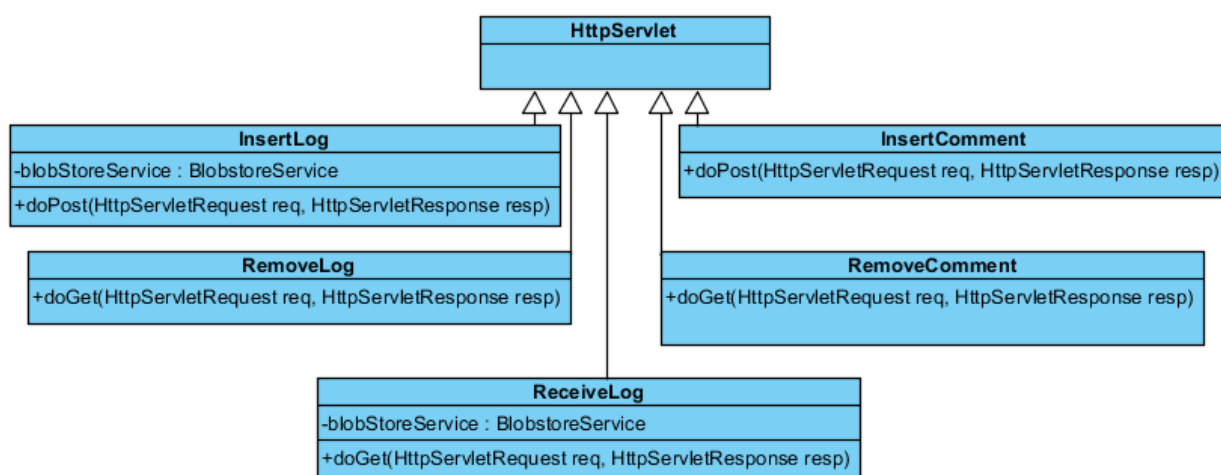


Figura 6.2: Classes do pacote *controls*.

6.3.3 Pacote Managers

O pacote *managers* (ver Figura 6.3) é formado por classes estáticas EMF (Entity Manager Factory) e PMF (Persistence Manager Factory). Essas classes são auxiliares e ajudam na criação de Logs e Comments persistentes.

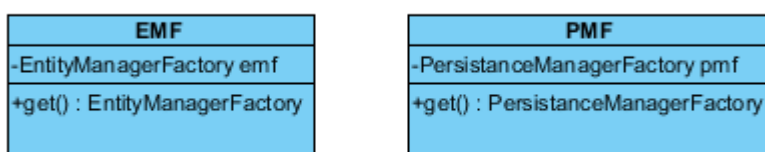


Figura 6.3: Classes do pacote *managers*.

6.4 Operação de inserir um Log

Abaixo, mostramos como ocorre o funcionamento interno da aplicação a partir da interação do usuário até o registro de suas informações de forma persistente.

Inicialmente, o usuário preenche um formulário como o da figura (6.4) abaixo:

Insert a new Log:

Issue: Tag:

Log:

Upload Nenhum arquivo selecionado

Figura 6.4: formulário para inserir um *log*.

Quando o preenchimento é concluído, significa que o botão “Create Log”, visto na Fig. # foi pressionado. Tal botão, quando clicado aciona o fragmento de código abaixo, que indica a criação de um novo log.

```
<form action="<%= blobstoreService.createUploadUrl("/new") %>"
```

O fragmento de código anterior continha como parâmetro o indicativo de uma nova (“/new”) criação de *Log*. Nesse ponto, o sistema faz consulta a lista de *Servlets* para encontrar qual a classe que implementa tal serviço. Como podemos observar no final do fragmento de código abaixo, tal serviço é implementado na classe *InsertLog*.

```
<servlet>
  <servlet-name>InsertLog</servlet-name>
  <servlet-class>game.log.controls.InsertLog</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>InsertLog</servlet-name>
  <url-pattern>/new</url-pattern>
</servlet-mapping>
```

Na classe *InsertLog* há o método *doPost()* que é responsável por coletar todas as informações (conteúdo do *Log*, *tag*, etc.) digitadas pelo usuário no formulário HTML.

```
public void doPost(HttpServletRequest req, HttpServletResponse resp)
    throws IOException {

    UserService userService = UserServiceFactory.getUserService();
    User author = userService.getCurrentUser();

    String content = req.getParameter("content");

    Text contentText = new Text(content);

    String tag = req.getParameter("tag");

    String issue = req.getParameter("issue");
```

Essas informações são organizadas e agrupadas para criação do Log, exibido no fragmento de código abaixo. O mesmo é armazenado persistentemente no sistema e por fim há o redirecionamento para a página principal do sistema atualizada com os dados que acabaram de ser postados.

```
//insert log
Log log = new Log(author, tag, issue, jiraTaskNumber,
    contentText, blobKey, blobID, filename, filenameExtension);
PersistenceManager pmLog = PMF.get().getPersistenceManager();
try {
    pmLog.makePersistent(log);
} finally {
    pmLog.close();
}

resp.sendRedirect("/main.jsp");
```

Os outros requisitos foram implementados seguindo o mesmo modelo e sua funcionalidade é análoga a apresentada. De forma geral o fluxo das informações (ver Figura 6.5) tem início com uma ação do usuário, que é identificada pelo sistema por meio dos *Servlets*, estes mapeiam qual classe (do pacote *controls*) implementa o serviço. Tais classes acessam, recuperam, criam ou modificam instâncias das classes do pacote *models* (*Log* e *Comment*) usando serviços das classes do pacote *managers* (EMF e PMF), bem como os serviços já implementados pelo *DatastoreService* disponível na distribuição da *App Engine*. Por fim, a resposta é enviada para o usuário por meio de uma atualização que exhibe o resultado da modificação do sistema.

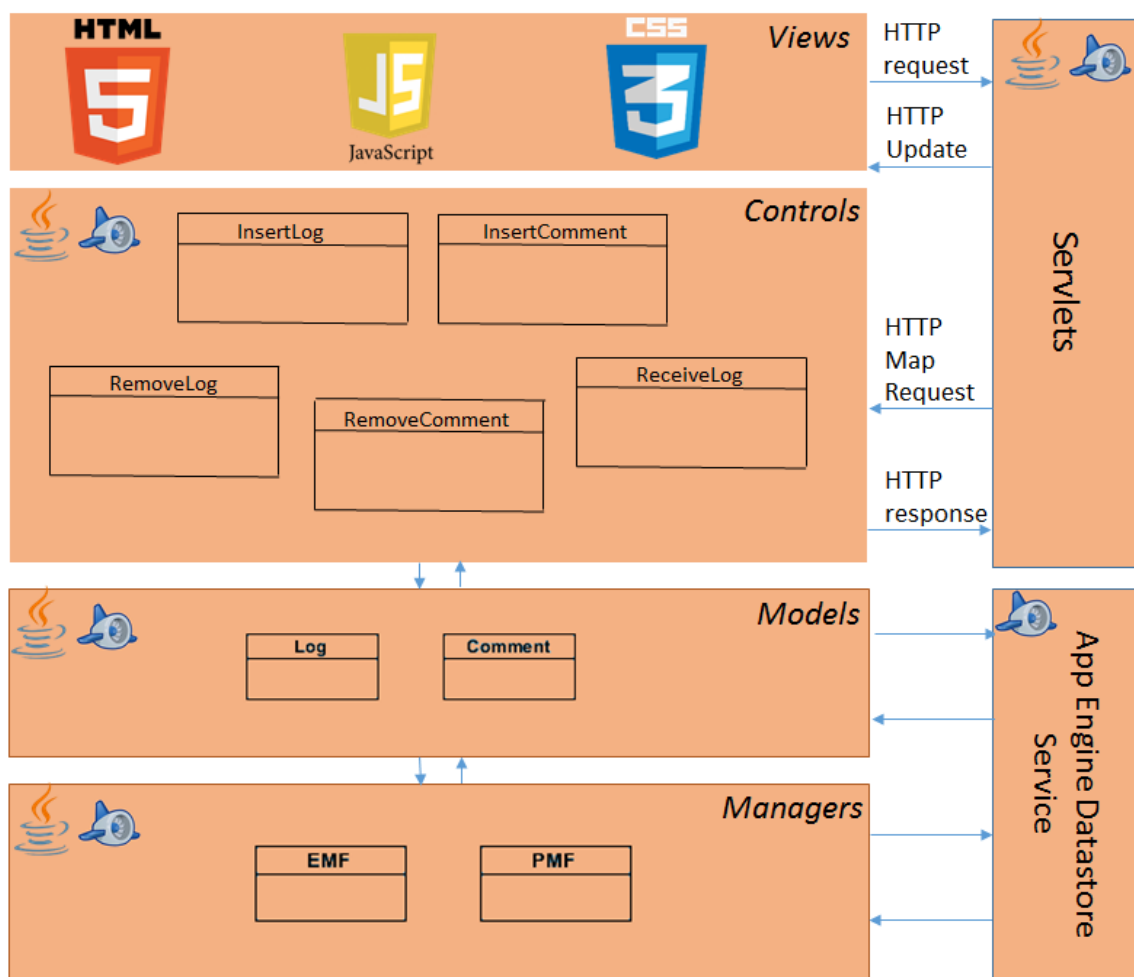


Figura 6.5: fluxo das informações no sistema. A interface com o usuário (*Views*) foi escrita em HTML5, JavaScript e CSS compreendendo o lado *cliente* do sistema. Todas as funcionalidades do lado *servidor* foram implementadas em *Java* com as bibliotecas de *Google App Engine*.

7 Conclusões

Neste trabalho investigamos os conflitos entre a pré-produção e a produção de games, mais precisamente nos concentramos nos conflitos gerados pelo GDD, artefato que está no coração da transição entre as fases citadas, mas que por uma série de razões não cumpre seu papel de orientar as equipes. Utilizamos uma série de métodos para entender tal problema, de questionários a entrevistas, consultamos profissionais com passagem por várias empresas do setor, tanto no Brasil quanto no exterior, que possuem experiência tanto na produção de games casuais para empresas com poucos anos no mercado, quanto na produção de games AAA para algumas das mais conhecidas e renomadas empresas do setor. Boa parte desse conhecimento foi transformado em princípios para a construção de um protótipo que nos permitiu avaliar como contribuir para atenuar tais conflitos. O protótipo foi testado pela Manifesto, uma empresa do porto digital com experiência de oito anos na produção de games de sucesso para os segmentos *web* e *mobile*. Nas várias semanas nas quais o protótipo foi testado, os funcionários da empresa nos forneceram valiosas críticas e sugestões de como melhorá-lo, bem como contribuíram para a discussão de como evitar que a passagem da pré-produção para a produção de games continue a ser tão conflituosa.

7.1 Contribuições

Dessa forma, acreditamos que as principais contribuições deste trabalho são as apresentadas abaixo.

7.1.1 Crítica ao GDD

Para definição do problema, fizemos inicialmente um esforço que considerou a aplicação de um questionário baseado em apontamentos de membros da indústria. O questionário foi respondido por aproximadamente 40 pessoas, a maioria, exercia cargos como os de Game Designers, Artistas ou Programadores (existindo líderes técnicos de cada uma dessas áreas, além de produtores) e responderam acerca de seus problemas em relação as fases de Design e Desenvolvimento. Como resultado, foi gerado uma lista de recomendações (*guidelines*) para ajudar desenvolvedores a evitar tais problemas. Identificamos que o GDD está no coração de muitos dos problemas listados, que é por

muitas vezes ignorado e considerado inadequado, embora exista uma grande quantidade de trabalhos destinada a conceitua-lo e elaborá-lo. Decidimos então, realizar outra pesquisa, baseada em entrevistas para ampliar a discussão anterior e identificar no discurso de profissionais da indústria quais realmente são os conflitos entre a pré-produção e a produção de games. Como resultado, os participantes continuaram a identificar o GDD como um artefato contraditório e insuficiente a cumprir aquilo a que se propõe, divulgaram suas formas alternativas de documentação, novas práticas, referências e suas opiniões para atenuar os problemas causados pela presença de um artefato que é difícil de escrever, difícil de manter, não atende aos públicos que necessitam de orientações para a produção e é consequentemente tratado com irrelevância, causando conflitos de comunicação, sobrecarga e retrabalho. Pela pequena quantidade de estudos que criticam diretamente o GDD, confiamos que tais afirmações apresentam contribuição ao tema.

7.1.2 Criação do Game Live Logs

De acordo com o discurso dos participantes, definimos princípios baseados em seus conhecimentos sobre como deve ser uma documentação de um projeto de games. Esse conhecimento deu origem a um *software* que decidimos chamar de **Game Live Logs**, por ser uma extensão do conceito de *Game Design Logs*, do Daniel Cook e por tornar o documento “vivo”, como desejou um dos colaboradores da pesquisa durante a fase de entrevistas.

7.1.3 Valorização da Análise e Observação do Problema para o Contexto Computacional

Consultamos a literatura diversas vezes, realizamos um questionário e diversas entrevistas até ter dados suficientes para propor uma solução. O conhecimento obtido foi organizado como princípios para uma nova forma de documentar games, os mesmos foram transformados em requisitos que deram origem ao protótipo do Game Live Logs. A experiência adquirida é que quanto mais abrangente for o estudo do domínio do problema, mais chances de sucesso terá a solução desenvolvida. Portanto, consideramos de grande valor que investimentos assim sejam feitos nas fases iniciais do desenvolvimento de software. Com os resultados obtidos na avaliação, percebemos

aceitação dos envolvidos em continuar usando o conceito desenvolvido, ainda que haja muitas melhorias a serem feitas. Caso não fizéssemos um estudo como o apresentado, muito provavelmente teríamos sugerido uma proposta de documentação mais tradicional como muitos já publicaram e continuam a publicar [Hugo, A. M. 2012], implicando nas mesmas desvantagens mencionadas e na busca de um GDD único e perfeito, o qual não acreditamos (pelos resultados desse trabalho) ser possível.

7.2 Trabalhos Futuros

Inicialmente devemos continuar a evolução do protótipo na Manifesto, bem como iniciar testes em outras empresas. No momento conseguimos interesse de mais uma que deve alocar uma equipe para tal tarefa assim que a mesma iniciar o desenvolvimento de um novo game. Dessa forma, com o aumento do número de usuários pretendemos avaliar a evolução do protótipo de forma quantitativa e compreender, a partir de uma amostragem maior, quais as prioridades para a manutenção do serviço. Em termos de implementação devemos criar o chat e o sistema de alertas, bem como criar uma versão para ser usada em *smartphones* e *tablets* permitindo cada vez mais que o sistema esteja sempre disponível. Seguindo a recomendação dos *Usuários Afoitos*, vamos implementar um meio de disponibilizar os *Logs* de forma impressa, consultar especialistas em UX e melhorar a interface gráfica e a usabilidade. Além das novas funcionalidades será preciso refatorar algumas sessões do código, sobretudo na interface. Muito da implementação foi realizada usando estruturas chamadas *scriptlets* (nos códigos em JSP), as quais descobrimos ao longo do projeto se tratar de uma prática em desuso que, embora não cause problemas de performance, possui problemas de legibilidade. O mais indicado é substituir os *scriptlets* por estruturas como *taglibs* (como o JSTL - JSP Standard Tag Library) ou *Expression Languages*. Nesse sentido, também consideramos importante a criação de um *plugin* para sistemas como o *Redmine* ou *Jira*, pela vantagem de usar os recursos já criados e a vasta comunidade de usuários. Além disso, as novas versões do Game Live Logs devem usar linguagens como *Python* ou *Ruby*, por consequência de sua proximidade com os sistemas anteriormente citados e por sua capacidade de adaptação ao dinamismo da *web*. Pretendemos lançar, assim que possível, o projeto para a comunidade de forma gratuita e com código aberto. Para isso devemos publicar no Google Code [2012], Sourceforge [2012] ou Git [2012] e dessa forma angariar novos colaboradores que possam ampliar o sistema de acordo com suas necessidades particulares e de seus times de

desenvolvimento. Um outro investimento na pesquisa é quanto à forma e conteúdo dos *Logs*. Nos foi bem claro o positivo impacto que a disponibilidade de tratar diversos conteúdos traz para as equipes de desenvolvimento, enquanto se concentram no design do game. Prover cada vez mais facilidade no armazenamento e nas formas de visualização desses conteúdos deve continuar a ser um dos principais objetivos deste trabalho.

Referências

- Alves, C. Ramalho, G. and Damasceno, A. *Challenges in Requirements Engineering for Mobile Games Development: The Meantime Case Study*, 2012 20th IEEE International Requirements Engineering Conference (RE), pp. 275-280, 15th IEEE International Requirements Engineering Conference (RE 2007), 2007.
- Amazon AWS. 2013. Disponível em: <aws.amazon.com>. Acesso em: junho 2013]
- Atlas.ti. 2012. Disponível em: <<http://www.atlasti.com>> . Acesso em: 10 junho 2013
- Axure. 2011. Disponível em: <www.axure.com>. Acesso em: 10 junho 2013
- Barros, R. 2009. *Investigando equipes Multidisciplinares em projetos de jogos: uma pesquisa qualitativa*. UFPE, Recife (2009)
- Bates, B. and Robert, A. 2004. *Game Design, 2nd Edition*. Course Technology Press, Boston, MA, United States.
- Bauer, M.W.. and Gaskell, G., 2000. *Qualitative researching with text, image and sound - a practical handbook*, London: Sage
- Blow, J. 2004. *Game Development: Harder Than You Think*. *Queue* 1, 10 (February 2004), 28-37.
- Bugzilla, 2009, Disponível em : <<http://www.bugzilla.org>>. Acessado em: 10 Novembro 2012.
- Buyya, R., Shin Yeo, C., Venugopal, S., Broberg, J. and Brandic, I. 2009. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.* 25, 6 (June 2009), 599-616.

- Calegario, F.C.A. *Sketchument: ambiente de experimentação para criação de instrumentos musicais digitais. Dissertação de Mestrado*. Pós Graduação em Ciência da Computação. Centro de Informática. Universidade Federal de Pernambuco. Recife. 2013
- Callele,D., Neufeld, E. and Schneider, K. 2005. Requirements Engineering and the Creative Process in the Video Game Industry. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering* (RE '05). IEEE Computer Society, Washington, DC, USA, 240-252.
- Callele,D., Neufeld, E. and Schneider, K. 2006. Emotional Requirements in Video Games. In *Proceedings of the 14th IEEE International Requirements Engineering Conference* (RE '06). IEEE Computer Society, Washington, DC, USA, 292-295.
- Callele,D., Neufeld, E. and Schneider, K. 2009. Visualizing Emotional Requirements. In *Proceedings of the 2009 Fourth International Workshop on Requirements Engineering Visualization* (REV '09). IEEE Computer Society, Washington, DC, USA, 1-10.
- Carvalho, G. *Uma Metodologia Preditiva para o Desenvolvimento de Jogos de Computador*. Trabalho de Graduação. Graduação em Ciência da Computação. Centro de Informática. Universidade Federal de Pernambuco. Recife. 2006.
- Cook, D., 2011. *Game Design Logs*. Lost Garden. Disponível em: <http://www.lostgarden.com/2011/05/game-designlogs.html>. Acesso em: 15 Jul. 2012.
- Dow, S. 2011. How prototyping practices affect design results. *Interactions* 18, 3 (May 2011), 54-59.
- Eclipse, 2012. Eclipse. Disponível em: <http://www.eclipse.org/>. Acesso em: 10 jun. 2013

- Falcão, T. P. R. and Gomes, A. S. *Modelagem de soluções ubíquas para uso em salas de aula do ensino fundamental*. In: GCETE 2004 - Global Congress on Engineering and Technology Education, 2004, São Vicente. Anais do GCETE'2004, 2004.
- Fernandez, L., Robles, S., Fortier, A., Ducasse, S., Rossi, G. and Gordillo, S. 2009. Meteoroid towards a real MVC for the web. In *Proceedings of the International Workshop on Smalltalk Technologies (IWST '09)*. ACM, New York, NY, USA, 28-37.
- Flick, U. *Uma introdução à pesquisa qualitativa*. 2.ed. Porto Alegre: Bookman, 2004. 312 p
- Flynt, J.P., 2004. *Software Engineering for Game Developers*, Muska and Lipman Publishing.
- GameDev.Net, 2009, Disponível em <http://www.gamedev.net/reference/articles/article1940.asp>.>. Acesso em: 10 Jun. 2013.
- Garcia, V. C.; Almeida, E. S.; Meira, S. R. M. *RiSE Reference Model for Software Reuse Adoption in Brazilian Companies*. In: Concurso de Teses e Dissertações em Qualidade de Software (CTDQS) X Simpósio Brasileiro de Qualidade de Software, 2011, Curitiba. Anais do Concurso de Teses e Dissertações em Qualidade de Software (CTDQS) X Simpósio Brasileiro de Qualidade de Software, 2011
- Git, 2012. Disponível em: <http://git-scm.com/> >. Acesso em: 10 jun. 2013
- Google App Engine, 2012. Disponível em: <https://cloud.google.com/products/>>. Acesso em: 10 jun. 2013
- Google Code, 2012. Disponível em: <https://code.google.com/>>. Acesso em: 10 jun. 2013.

Hirsch, E., Stringfellow, R., Amer, P., Goodrich, B., Marshall, J., and Cliff Brett, L. 2007. *Crossing the line: Moving from film to games and possibly back*. In *ACM SIGGRAPH 2007 courses* (SIGGRAPH '07). ACM, New York, NY, USA, 1-94.

Highlight.js, 2012. SoftwareManiacs. Disponível em: <http://softwaremaniacs.org/soft/highlight/en/> >. Acesso em: 10 jun. 2013

Hugo A. M. 2012. Proposal of Game Design Document from software engineering requirements perspective. In *Proceedings of the 2012 17th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES)* (CGAMES '12). IEEE Computer Society, Washington, DC, USA, 81-85.

Isbister, K. and Schaffer, N. 2008. *Game Usability: Advancing the Player Experience*. Morgan Kaufmann Elsevier

Jira, 2012. Atlassian. Disponível em: www.atlassian.com/software/jira Acesso em: 10 jun. 2013

Kellison, C., 2008. *Producing For TV and New Media, Second Edition: A Real World Approach for Producers*. FocalPress second.

Kujala, S., Kauppinen, M., and Rekola, S. (2001). *Bridging the gap between user needs and user requirements*. In Avouris, N. and Fakotakis, N., editors, *Advances in Human-Computer Interaction I* (Proceedings of the Panhellenic Conference with International Participation in Human-Computer Interaction PC – HCI 2001), pages 45–50. Typorama Publications.

Kujala, S. and Kauppinen, M. 2004. Identifying and selecting users for user-centered design. In *Proceedings of the third Nordic conference on Human-computer interaction* (NordiCHI '04). ACM, New York, NY, USA, 297-303.

- Kuuskeri, J. 2011. Experiences on a design approach for interactive web applications. In *Proceedings of the 2nd USENIX conference on Web application development* (WebApps'11). USENIX Association, Berkeley, CA, USA, 8-8.
- Lang, T., 2009. *Four Ways to Write Your Design Docs*. Game career guide, p.3. Disponível em: <http://www.gamecareerguide.com/features/737/four_ways_to_write_your_design_.php?page=1> Acesso em: 9 Jun. 2012
- L.A. Noire. 2011. Rockstar Disponível em: <<http://www.rockstargames.com/lanoire/agegate/ref/?redirect=>>>. Acesso em: 10 jun. 2013
- Li, A., Yang, X., Kandula, S. and Zhang, M. 2010. CloudCmp: comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (IMC '10). ACM, New York, NY, USA, 1-14.
- Machado, T.L.A., 2009. *Guidelines Para a Criação de Jogos: Boas Práticas Para Reduzir Conflitos Entre o Design e o Desenvolvimento*. Universidade Federal de Pernambuco.
- Machado, T.L.A., Ramalho, G.L. and Alves, C., 2011. *Games as Cinema*. In: X Brazilian Symposium on Games and Digital Entertainment, 2011, Salvador, Bahia. Brazil.
- Machado, T. L. A.; Ramalho, G. L.; Alves, C ;Garcia, V C; Lemos, V.; Araujo, L. F.; Fabrino, V. G.; Silva, A. P; Xavier, B. *Game Development Guidelines: Practices To Avoid Conflicts Between Software and Design*. In: IX Brazilian Symposium on Games and Digital Entertainment, 2010, Florianópolis, Santa Catarina. Brazil.
- Machado, T. L. A., Lima, R., Florencio, F., Ramalho, G. and Alves, C. *The Game Development Conflicts According to the Game Industry*. In: XI Brazilian Symposium on Games and Digital Entertainment, 2012, Brasília DF, Brazil.

- Maruya, A. 2012. *Running Lean: Iterate from Plan A to a Plan That Works*. O' Reilly Media; Second Edition edition (March 6, 2012).
- Manifesto Games, 2012. Disponível em: www.manifestogames.com.br. Acesso em: 10 Jun. 2012
- Magenheim, J., Nelles, W., Rhode, T., Schaper, N., Schubert, S., and Stechert, P. 2010. *Competencies for informatics systems and modeling: Results of qualitative content analysis of expert interviews*. In IEEE EDUCON 2010 Conference. IEEE, 513-521.
- Mayring, P., 2004. *Qualitative Content Analysis*, U. Flick, E.v. Kardoff and I. Steinke (eds), A Companion to Qualitative Research, London: SAGE.
- Merriam, S.B., 2009. *Qualitative research: a guide to design and implementation*, San Francisco: Jossey-Bass
- Meuser, M., Nagel, U.: *ExpertInneninterviews - vielfach erprobt, wenig bedacht Ein Beitrag zur qualitativen Methodendiskussion* In: Bogner, A., Littig, B., Menz, W (eds.) *Das Experteninterview Theorie, Methode, Anwendung*, pp 71-93 Leske und Budrich, Opladen (2002)
- Mustapic, G., Wall, A., Norström, C., Crnkovic, I., Sandström, K., Fröberg, J. and Andersson, J. *Real world influences on software architecture - interviews with industrial system experts*, IEEE/IFIP Conference on Software Architecture, pp. 101-112, 2004.
- Myllyaho, M., Salo, O., Kääriäinen, J., Hyysalo, J., and Koskela, J. 2004. *A review of small and large postmortem analysis methods*. In Proceedings of the IEEE 17th International Conference on Software and Systems Engineering and their Applications.
- Perry, D. *David Perry On Game Design*. 2009. Cengage Learning; 1 edition (March 24, 2009)

- Petrillo, F., Pimenta, M., Trindade, F. and Dietrich, C. 2008. Houston, we have a problem...: a survey of actual problems in computer games development. In *Proceedings of the 2008 ACM symposium on Applied computing* (SAC '08). ACM, New York, NY, USA, 707-711.
- Petrillo, F., Pimenta, M., Trindade, F. and Dietrich, C. 2009. What went wrong? A survey of problems in game development. *Comput. Entertain.* 7, 1, Article 13 (February 2009), 22 pages.
- Petrillo, F. and Pimenta, M. 2010. Is agility out there?: agile practices in game development. In *Proceedings of the 28th ACM International Conference on Design of Communication* (SIGDOC '10). ACM, New York, NY, USA, 9-15.
- phpBB, 2012. Disponível em: <<https://www.phpbb.com/>> Acesso em: 10 jun. 2013
- Potanim, R., 2010. *Forces in play*. In Proceedings of the 3rd International Conference on Fun and Games - Fun and Games '10. New York, New York, USA: ACM Press, pp. 135–143.
- Rollings, A. and Morris, D., 1999. *Game Architecture and Design*, Coriolis Group Books.
- Rouse III, R. 2004. *Game Design: Theory and Practice*. Wordware Publishing Inc., Plano, TX, USA.
- Runeson, P. and Host, M., 2008. *Guidelines for conducting and reporting case study research in software engineering*. Empirical Software Engineering, 14th ed.
- Santos, H. *Fatores Críticos de Sucesso Das Iniciativas de BPM no Setor Público*. Dissertação de Mestrado, Pós - Graduação em Ciência da Computação. Centro de Informática. Universidade Federal de Pernambuco. 2012.

- Schuytema, P. 2006. *Game Design: A Practical Approach (Game Development Series)*. Charles River Media, Inc., Rockland, MA, USA.
- Sharp, H., Rogers, Y. and Preece, J. 2007. *Interaction Design: Beyond Human Computer Interaction*. John Wiley & Sons.
- Sheffield, B., 2009. *What Went Wrong? Learning From Past Postmortems*. Disponível em:
http://www.gamasutra.com/view/feature/4001/what_went_wrong_learning_from_.php.> Acesso em: 10 Jun 2013.
- Schell, J., 2008. *The Art of Game Design: A Book of Lenses*. San Francisco: Morgan Kaufmann Publishers Inc.
- Sommerville, I. And Sawyer, P. *Requirements Engineering: A Good Practice Guide*. Wiley (April 28, 1997).
- Sourceforge, 2012. Disponível em: <http://sourceforge.net/> Acesso em: 10 jun 2013
- Souza, L.J.E.A., 2009. *Análises de Documento de Game Design: Interpretação e Resultados Gerados*. In VIII Brazilian Symposium on Games and Digital Entertainment. Rio de Janeiro: PUC – Rio
- Stäbler, A. 2008. A service oriented loosely coupled GUI framework in the mobile context. In *Proceedings of the 2008 conference on Techniques and Applications for Mobile Commerce: Proceedings of TAMoCo 2008*, Cherif Branki, Brian Cross, Gregorio Díaz, Peter Langendörfer, Fritz Laux, Guadalupe Ortiz, Martin Randles, A. Taleb-Bendiab, Frank Teuteberg, Rainer Unland, and Gerhard Wanner (Eds.). IOS Press, Amsterdam, The Netherlands, The Netherlands, 64-76.
- Thomke, S. (2003). *Experimentation Matters: Unlocking the Potential of New Technologies for Innovation*. Harvard Business School Press Books. Harvard Business School Press.

Wang, X., Conboy, K. and Cawley, O. 2012. *"Leagile" software development: An experience report analysis of the application of lean approaches in agile software development* .J. Syst. Softw. 85, 6 (June 2012), 1287-1299.

Widdgeditor, 2012. Disponível em: <<https://code.google.com/p/widgeditor/>> Acesso em: 10 jun 2013

Windows Azure, 2012. Disponível em: <www.windowsazure.com> Acesso em: 10 jun 2013

Winget, M.A. and Sampson, W.W., 2011. *Game development documentation and institutional collection development policy*. In Proceeding of the 11th annual international ACM/IEEE joint conference on Digital libraries - JCDL '11. New York, New York, USA: ACM Press, p. 29.

Yin, R.K., 2008. *Case Studie Research – Design And Methods* 4th ed., SAGE PUBLICATIONS

NOTA: as figuras 5.5 e 5.11 são respectivamente do game Ski Safari ® [http://www.defiantdev.com/ski-safari.php] e do personagem Mario, da Nintendo ® Ambas foram usadas apenas para propósitos ilustrativos. A imagem da página 70 é de um vídeo disponível em um site de arquivos multimídia para testes e pode ser encontrado no link [http://archive.org/details/Pbtestfilemp4videotestmp4].

ANEXO 1 – Protocolo das Entrevistas

O quadro a seguir resume nossos objetivos com este estudo.

Analisar	As etapas de pré-produção e produção de games;
Com a proposta de	Encontrar quais as necessidades que mais afligem os desenvolvedores (artistas, designers, programadores, produtores);
A partir do ponto de vista	Dos produtores, líderes técnicos (software), programadores e artistas;
No seguinte contexto	Produções de games dos participantes da pesquisa.

1. Roteiro das Entrevistas

A Seguir são apresentados os roteiros de entrevista para cada um dos membros dos grupos definidos: Produtor / Gerente de projetos, Líder técnico software (programador), Game Designer e Artista ou Líder técnico em artes . As perguntas foram definidas tendo em base nossas pesquisas [Machado, T.L.A., 2009; Machado, T. L. A. et al 2010; Machado, T. L. A. et al 2011], bem como nossa revisão de literatura.

- Produtor / Gerente de Projetos

Questões Principais	Questões Secundárias
1. Quais são as suas maiores necessidades em pré-produção?	1.1. O que realmente ocorre durante sua pré-produção? 1.2. Quais problemas são recorrentes durante sua pré-produção? 1.3. O que você sente mais falta na pré-produção? 1.4. O que você considera que pode deixar sua pré-produção mais efetiva?
2. Como ocorre a comunicação entre fases (pré-produção e produção)?	2.1. Ela é feita através de artefatos? Quais?

	<p>2.2. Você a considera eficiente, os desenvolvedores realmente entendem o que foi decidido?</p> <p>2.3. O que mais afeta seu trabalho nessa comunicação?</p> <p>2.4. O que você acha que poderia melhorar tal comunicação?</p>
<p>3. Como são definidos o Escopo do Projeto do Game?</p>	<p>3.1. Quais técnicas/ferramentas você utiliza para definição de Escopo?</p> <p>3.2. Você considera que o escopo é atingindo durante a produção?</p> <p>3.3. Quais as maiores necessidades que você enfrenta quando está definindo um escopo?</p> <p>3.3.1. Quem lhe ajuda nessa tarefa?</p> <p>3.3.2. O cliente participa?</p> <p>3.3.3. Como você estima o tempo de produção e os recursos necessários?</p> <p>3.4. O que você acha que melhoraria tal processo de definição?</p>
<p>4. Quais documentações técnicas você utiliza?</p>	<p>4.1. Em que momento elas são produzidas?</p> <p>4.2. Como são apresentadas?</p> <p>4.3. Qual a importância que elas trazem para a produção?</p> <p>4.4. Como a equipe de produção as encara?</p> <p>4.5. O que lhe incomoda em tais documentações?</p> <p>4.6. O que você considera que pode trazer melhorias?</p>

<p>5. Quais suas maiores necessidades durante a produção?</p>	<p>5.1. O que realmente você faz durante a produção?</p> <p>5.2. Você utiliza alguma ferramenta para lhe auxiliar durante a produção?</p> <p>5.3. Como as definições da pré-produção lhe ajudam na produção?</p> <p>5.4. O que você acha que pode melhorar a produção?</p>
<p>6. Caso você tivesse um super computador, capaz de produzir qualquer tipo de game apenas ouvindo você, quais informações você daria para ele?</p>	<p>6.1. Se você pudesse automatizar qualquer parte do seu trabalho, qual seria?</p>

- Líder Técnico – Software e Programador

Questões Principais	Questões Secundárias
<p>1. Quais são as suas maiores necessidades em pré-produção?</p>	<p>1.1. O que realmente ocorre durante sua pré-produção?</p> <p>1.2. O que é essencial definir durante a pré-produção?</p> <p>1.3. Quais problemas são recorrentes durante sua pré-produção?</p> <p>1.3 O que você sente mais falta na pré-produção?</p> <p>1.4. O que você considera que pode deixar sua pré-produção mais efetiva?</p>
<p>2. Como ocorre a comunicação entre fases (pré-produção e produção)?</p>	<p>2.1. Ela é feita através de artefatos? Quais?</p> <p>2.2. Você a considera eficiente, os desenvolvedores realmente entendem o que foi decidido?</p>

	<p>2.3. O que mais afeta seu trabalho nessa comunicação?</p> <p>2.4. O que você acha que poderia melhorar tal comunicação?</p> <p>2.5. Como é sua comunicação com o Game Designer e com os Artistas?</p> <p>2.5.1 O que discutem?</p> <p>2.5.2 Com que frequência se encontram?</p>
3. Como são definidos o Escopo do Projeto do Game?	<p>3.1. Quais técnicas/ferramentas você utiliza para definição de Escopo?</p> <p>3.2. Você considera que o escopo é atingindo durante a produção?</p> <p>3.3. Quais as maiores necessidades que você enfrenta quando está definindo um escopo?</p> <p>3.3.1. Quem lhe ajuda nessa tarefa?</p> <p>3.3.2. O cliente participa?</p> <p>3.3.3. Como você estima o tempo de produção e os recursos necessários?</p> <p>3.4. O que você acha que melhoraria tal processo de definição?</p>
4. Quais documentações técnicas você utiliza?	<p>4.1. Em que momento elas são produzidas?</p> <p>4.2 Como são apresentadas?</p> <p>4.3. Qual a importância que elas trazem para a produção?</p> <p>4.4. Como a equipe de produção as encara?</p> <p>4.5. O que lhe incomoda em tais documentações?</p> <p>4.6. O que você considera que pode trazer melhorias?</p>
5. Quais suas maiores necessidades durante a produção?	<p>5.1. O que realmente você faz durante a produção?</p>

	<p>5.2. Você utiliza alguma ferramenta para lhe auxiliar durante a produção?</p> <p>5.3. Como as definições da pré-produção lhe ajudam na produção?</p> <p>5.4. O que você acha que pode melhorar a produção?</p>
6. Como é sua integração com o Design do Game?	<p>5.1. Você está integrado desde a pré-produção?</p> <p>5.2. Como tal integração afeta sua produção?</p> <p>5.3. O que você considera essencial em um Design de Game para sua produção.</p> <p>5.4. O que você acha que pode melhorar tal integração?</p>
7. Caso você tivesse um super computador inteligente, capaz de produzir qualquer tipo de game, quais informações você daria para ele?	<p>7.1 Se você pudesse automatizar qualquer parte do seu trabalho, qual seria?</p>

- **Game Designer**

Questões Principais	Questões Secundárias
<p>1. Quais são as suas maiores necessidades em pré-produção?</p>	<p>1.1. O que realmente ocorre durante sua pré-produção?</p> <p>1.2 O que é essencial definir durante a pré-produção?</p> <p>1.3. Quais problemas são recorrentes durante sua pré-produção?</p> <p>1.4. O que você mais sente falta durante a pré-produção?</p>

	<p>1.5. O que você considera que pode deixar sua pré-produção mais efetiva?</p>
<p>2. Como ocorre a comunicação entre fases (pré-produção e produção)?</p>	<p>2.1. Ela é feita através de artefatos? Quais?</p> <p>2.2. Você a considera eficiente, os desenvolvedores (não só programadores) realmente entendem o que foi decidido?</p> <p>2.3. O que mais afeta seu trabalho nessa comunicação?</p> <p>2.4. O que você acha que poderia melhorar tal comunicação?</p> <p>2.5. Como é sua comunicação com os times (Programação, Arte, Design)?</p> <p>2.5.1 O que discutem?</p> <p>2.5.2 Com que frequência se encontram?</p>
<p>3. Como são definidos o Escopo do Projeto do Game?</p>	<p>3.1. Quais técnicas/ferramentas você utiliza para definição de Escopo?</p> <p>3.2. Você considera que o escopo é atingindo durante a produção?</p> <p>3.3. Quais as maiores necessidades que você enfrenta quando está definindo um escopo?</p> <p>3.3.1. Quem lhe ajuda nessa tarefa?</p> <p>3.3.2. O cliente participa?</p> <p>3.3.3. Como você estima o tempo de produção e os recursos necessários?</p> <p>3.4. O que você acha que melhora tal processo de definição?</p>
<p>4. Quais documentações técnicas você utiliza?</p>	<p>4.1. Em que momento elas são produzidas?</p> <p>4.2 Como são apresentadas?</p>

	<p>4.3. Qual a importância que elas trazem para a produção?</p> <p>4.4. Como a equipe de produção as encara?</p> <p>4.5. O que lhe incomoda em tais documentações?</p> <p>4.6. O que você considera que pode trazer melhorias?</p>
5. Quais suas maiores necessidades durante a produção?	<p>5.1. O que realmente você faz durante a produção?</p> <p>5.2. Você utiliza alguma ferramenta para lhe auxiliar durante a produção?</p> <p>5.3. Como as definições da pré-produção lhe ajudam na produção?</p> <p>5.4. O que você acha que pode melhorar a produção?</p>
6. Como é sua integração com os Programadores e Artistas?	<p>5.1. Você está integrado desde a pré-produção?</p> <p>5.2. Como tal integração afeta sua produção?</p> <p>5.3. O que você considera essencial em um Design de Game para sua produção.</p> <p>5.4. O que você acha que pode melhorar tal integração?</p>
7. Caso você tivesse um super computador inteligente, capaz de produzir qualquer tipo de game, quais informações você daria para ele?	<p>7.1 Se você pudesse automatizar qualquer parte do seu trabalho, qual seria?</p>

- Artista / Líder técnico (Artes)

Questões Principais	Questões Secundárias
1. Quais são as suas maiores necessidades em pré-produção?	<p>1.1. O que realmente ocorre durante sua pré-produção?</p> <p>1.2. O que é essencial definir durante a pré-produção?</p> <p>1.3. Quais problemas são recorrentes durante sua pré-produção?</p> <p>1.4. O que você mais sente falta durante a pré-produção?</p> <p>1.5. O que você considera que pode deixar sua pré-produção mais efetiva?</p>
2. Como ocorre a comunicação entre fases (pré-produção e produção)?	<p>2.1. Ela é feita através de artefatos? Quais?</p> <p>2.2. Você a considera eficiente, os artistas realmente entendem o que foi decidido?</p> <p>2.3. O que mais afeta seu trabalho nessa comunicação?</p> <p>2.4. O que você acha que poderia melhorar tal comunicação?</p> <p>2.5. Como é sua comunicação com os times (Programação, Design)?</p> <p>2.5.1 O que discutem?</p> <p>2.5.2 Com que frequência se encontram?</p>
3. Como são definidos o Escopo do Projeto do Game?	<p>3.1. Quais técnicas/ferramentas você utiliza para definição de Escopo?</p> <p>3.2. Você considera que o escopo é atingindo durante a produção?</p>

	<p>3.3. Quais as maiores necessidades que você enfrenta quando está definindo um escopo?</p> <p>3.3.1. Quem lhe ajuda nessa tarefa?</p> <p>3.3.2. O cliente participa?</p> <p>3.3.3. Como você estima o tempo de produção e os recursos necessários?</p> <p>3.4. O que você acha que melhora tal processo de definição?</p>
4. Quais documentações técnicas você utiliza?	<p>4.1. Em que momento elas são produzidas?</p> <p>4.2. Como são apresentadas?</p> <p>4.3. Qual a importância que elas trazem para a produção?</p> <p>4.4. Como a equipe de produção (de assets) as encara?</p> <p>4.5. O que lhe incomoda em tais documentações?</p> <p>4.6. O que você considera que pode trazer melhorias?</p>
5. Quais suas maiores necessidades durante a produção?	<p>5.1. O que realmente você faz durante a produção?</p> <p>5.2. Você utiliza alguma ferramenta para lhe auxiliar durante a produção?</p> <p>5.3. Como as definições da pré-produção lhe ajudam na produção?</p> <p>5.4. O que você acha que pode melhorar a produção?</p>
6. Como é sua integração com os Programadores e Artistas?	<p>5.1. Você está integrado desde a pré-produção?</p> <p>5.2. Como tal integração afeta sua produção?</p>

	<p>5.3. O que você considera essencial em um Design de Game para sua produção.</p> <p>5.4. O que você acha que pode melhorar tal integração?</p>
<p>7. Caso você tivesse um super computador inteligente, capaz de produzir qualquer tipo de game, quais informações você daria para ele?</p>	<p>7.1 Se você pudesse automatizar qualquer parte do seu trabalho, qual seria?</p>

2. Referências

<As mesmas listadas como referências deste trabalho>

ANEXO 2 – Roteiro das entrevistas com especialistas

1. Você acredita ser fácil de usar tal sistema?
2. Que melhorias você adicionaria?
3. Em que cenários você acha que o uso do sistema beneficiará suas atividades?
4. Você tem ou continua interessado em usar tal sistema?