
**Universidade Federal de Pernambuco**

**Centro de Informática**

**Ciência da Computação**

# Guidelines Para a Criação de Jogos: Boas Práticas Para Reduzir Conflitos Entre o Design e o Desenvolvimento

**Trabalho de Graduação**

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

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

**Co-Orientador(a):** Profª. Carina Frota Alves *(cfa@cin.ufpe.br)*

Recife, Dezembro de 2009

**Universidade Federal de Pernambuco**

**Centro de Informática**

**Ciência da Computação**

# Guidelines Para a Criação de Jogos: Boas Práticas Para Reduzir Conflitos Entre o Design e o Desenvolvimento

**Trabalho de Graduação**

Monografia apresentada ao Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do Grau de Bacharel em Ciência da Computação.

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

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

**Co-Orientador(a):** Prof[a]. Carina Frota Alves *(cfa@cin.ufpe.br)*

Recife, Dezembro de 2009

## Assinaturas

Este Trabalho de Graduação é resultado dos esforços do aluno Tiago Lemos de Araujo Machado, orientado pelo professor Geber Lisboa Ramalho, com o título "*Guidelines Para Criação de Jogos: Boas Práticas Para Reduzir Conflitos Entre o Design e o Desenvolvimento*". Todos abaixo estão de acordo com o conteúdo deste documento e os resultados deste Trabalho de Graduação

Orientador

_____

Professor Geber Lisboa Ramalho

Avaliador

_____

Professor André Menezes Marques das Neves.

Aluno

_____

Tiago Lemos de Araujo Machado

Recife, Dezembro de 2009

## 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 na UFPE 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 valiosa colaboração na minha Iniciação Cientifica.

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.

Aos professores Geber Ramalho, Carina Alves e André Neves por aceitarem colaborar com o desenvolvimento deste trabalho.

**Obrigado!**

## Dedicatória

***Àqueles que se esforçam em***
***imaginar, criar, divertir e emocionar***
***este trabalho é dedicado...***

*Através de grossas portas,*
*sentem-se luzes acesas,*
*— e há indagações minuciosas*
*dentro das casas fronteiras.*
*"Que estão fazendo, tão tarde?*
*Que escrevem, conversam, pensam?*
*Mostram livros proibidos?*
*Lêem notícias nas Gazetas?*
*Terão recebido cartas*
*de potências estrangeiras?"*
*(Antiguidades de Nimes*
*em Vila Rica suspensas!*
*Cavalo de La Fayette*
*saltando vastas fronteiras!*
*Ó vitórias, festas, flores*
*das lutas da Independência!*
*Liberdade - essa palavra,*
*que o sonho humano alimenta:*
*que não há ninguém que explique,*
*e ninguém que não entenda!)*
*E a vizinhança não dorme:*
*murmura, imagina, inventa.*
*Não fica bandeira escrita,*
*mas fica escrita a sentença.*

*Cecília Meireles*
*- Romanceiro da Inconfidência -*

**Resumo**

O processo de desenvolvimento de jogos é um trabalho multidisciplinar que envolve várias tarefas e fases. Esse processo é guiado tradicionalmente por um Documento de Game Design (DGD), tal documento contém as informações necessárias para que toda a equipe envolvida no processo possa desenvolver o Jogo.

Entretanto, na prática, não é o que ocorre para os encarregados da etapa de programação. Muitas das informações acabam sendo insuficientes ou acarretam em diversas interpretações, gerando um conflito entre a descrição do DGD e o Jogo em desenvolvimento.

Esse trabalho tem o objetivo de estudar os processos de desenvolvimento em andamento e analisar principalmente como ocorre a passagem da fase de finalização do Documento de Game Design ao inicio da implementação do software do Jogo.

**Palavras-chave:** entretenimento digital, jogos eletrônicos, processos e metodologias de desenvolvimento de software, game design

**Abstract**

The process of develop a game is multidisciplinary and enrolls several tasks and phases. This process is traditionally guide by the Game Design Document (GDD), a document which contains all the necessary info for all entire crew enrolled in the development process can develop the game.

Although, in really, it isn´t happen for the professionals of the program phases. Many contents becomes insufficient, starting a conflict between the description of the GDD and the game being developed.

This work has the purpose to study the development process and verify principally how is the ways of the end of a GDD and the start of the game software programming phase.

**Keywords:** digital entertainment, electronic games, process and software development methodologies, game design

## Sumário

# LISTA DE TABELAS

# LISTA DE GRÁFICOS

## LISTA DE GUIDELINES

# 1. Introdução

O processo de desenvolvimento de jogos digitais vem passando por várias mudanças ao longo dos tempos. De um modelo de produção puramente *ad hoc* e experimental, muitas vezes destinados a propósitos acadêmicos no seu inicio até a atualidade na qual são gerados produtos que passam por uma escala tida por muitos como cinematográfica.

A grande mudança de impacto nos processos de desenvolvimento de jogos ocorreu quando se passou a perceber que o nível de infra-estrutura dos equipamentos podia comportar jogos de maior qualidade audiovisual.

Dessa forma o mercado consumidor passou a exigir cada vez mais produtos com alto nível de sofisticação. Assim, os recursos das arquiteturas de hardware e software necessitavam de uma nova visão que adaptassem os recursos da produção de jogos a essa necessidade de mercado.

O modelo de produção deixava de ser visto apenas em função da capacidade de processamento o que implicou em jogos de apurado apelo artístico visual, de trilhas sonoras refinadas (contando em muitos casos com a participação de artistas conhecidos da indústria fonográfica), de personagens incorporados a cultura popular e de tramas por vezes tão complexas e intrincadas quanto a de grandes romances policiais.

A presença do Game Designer e sua capacidade de equilibrar todos esses elementos dentro de uma estrutura interativa, capaz de extrair diversão e emoção das pessoas, tornou-se fundamental à indústria de jogos. O que implicou em um modelo de produção que é regulado de acordo com um documento: o Documento de Game Design.

Entretanto, apesar de sua importância inquestionável, a tarefa de extrair informações do Documento de Game Design para obter o suporte à implementação do *software* que executará o jogo ainda é muito custosa dentro do processo atual de desenvolvimento. A preocupação em tornar esse intervalo mais curto torna-se de grande prioridade devido aos riscos, gastos e ganhos da indústria, nunca antes tão altos **[Calele, 2005]**.

Esse estudo dedicou-se a investigar como ocorre tal etapa dentro dos processos de jogos utilizados e na geração de *guidelines* com a intenção de apoiar na tomada de decisões por parte dos envolvidos na produção de jogos digitais.

## 1.1 Objetivos e Estrutura do Trabalho

### *1.1.1 Objetivo Geral*

O objetivo geral deste estudo é compreender como se dá atualmente o procedimento de tomada de decisões que definem o intervalo entre a definição de características do Design do Jogo e sua conseqüente implementação.

### *1.1.2 Objetivos Específicos*

Para atingir o objetivo geral, utilizamos as seguintes metas:

- Compreensão dos elementos tidos como padrão para um Documento de Game Design;

O Documento de Game Design é um artefato do processo de desenvolvimento de Jogos passível de mudanças que atendem a um grande número de fatores: seu (s) autor (es) e o estilo do Jogo a ser criado são alguns deles. Portanto para esse estudo, foi necessário definir uma padronização, a exemplo de **[Rodriguez, 2007]** de tal artefato para facilitar a comunicação com membros da indústria e tornar mais clara as dificuldades enfrentadas pelos mesmos para transformar tais informações nos softwares dos jogos.

- Identificar os problemas relacionados à interpretação do Documento de Game Design para desenvolvimento do software do jogo;

A partir de uma compreensão sobre os elementos padrão para um Documento de Game Design, verificar a partir de pessoas diretamente envolvidas com a produção de jogos quais os principais problemas enfrentados diariamente em converter tais informações contidas no artefato para gerar os softwares dos jogos.

- Gerar alternativas para solução desses problemas;

Analisar as informações obtidas sobre os problemas de interpretação do artefato para se obter a implementação do jogo, de forma a gerar alternativas que possam guiar a equipe de desenvolvimento a evitar tais dificuldades.

### *1.1.3 Estrutura do trabalho*

Este trabalho está estruturado da seguinte forma, no primeiro capitulo faz-se uma introdução sobre o problema a ser analisado e os trabalhos relacionados já publicados sobre temas semelhantes.

O segundo capitulo trata sobre o Documento de Game Design e os elementos que o compõe. O terceiro apresenta metodologias para o desenvolvimento de software e como elas têm sido adaptadas para os processos de desenvolvimento de jogos. No quarto há a apresentação da metodologia de trabalho, como os resultados apresentados no capitulo cinco.

O sexto e o sétimo capítulos são dedicados respectivamente as conclusões e aos trabalhos futuros, enquanto que o oitavo revela todas as referências utilizadas no projeto.

## 1.2 Trabalhos Relacionados

**[Rodriguez, 2007]:** este trabalho mostra uma abordagem interessante para o processo de desenvolvimento de jogos educativos. Podemos perceber que em tal processo, a correspondência entre o Documento de Game Design e a implementação já se inicia a partir do momento em que a equipe é selecionada. Os membros do time ganham o conhecimento a respeito do que é o projeto e combinam suas experiências na geração de soluções. Até o momento da publicação do trabalho três jogos haviam sido lançados. As especificações consideradas suficientes para a equipe de produção tornaram-se as seguintes etapas de desenvolvimento de software dos jogos do projeto:

- Identificação de componentes reusáveis
- Implementação de Interfaces Gráficas e funcionalidades do jogo.
- Avaliação: testes de integração, unidade e usabilidade

O processo de Design e de Desenvolvimento é iterativo e passível de mudanças a qualquer momento. A inspiração veio de modelos clássicos da Engenharia de Software, como Rational Unified Process - RUP e *Waterffal* (Cascata), porém adaptados para permitirem seu funcionamento de forma cíclica.

**[Calele, 2005]:** esse estudo trata da dificuldade de se coletar requisitos de Jogos em meio as informações contidas no Documento de Game Design. O autor cita que dependendo de quem é encarregado da atividade de analisar o documento, os requisitos elicitados podem ser variados. O estudo tomou como base uma sessão dedicada a tratar de problemas no desenvolvimento de Jogos da revista Game Developer **[Game Developer Magazine, 2009]**, Uma das alternativas propostas para diminuir problemas desse tipo é um investimento maior na fase de pré-produção do jogo, na qual seriam geradas todo um material de suporte prévio, como protótipos e arquiteturas que facilitariam a posterior identificação de requisitos.

**[Bergan, 2007]**: o livro traz uma introdução na qual apresenta as etapas do processo de produção cinematográfica. Assim como na indústria de jogos, a indústria de filmes também tem seus processos regulados de acordo com um documento **[Field, 2001]**: o roteiro.

A passagem das informações descritas no roteiro para o que é visto na tela ocorre com o auxilio de uma etapa de pré-produção, defendida como útil no desenvolvimento de jogos em **[Calele, 2005],** na qual são gerados vários *storyboards*, *art concepts* e pré-visualizações que permitem ao diretor e equipe de produção avaliar as melhores maneiras para registrar cada uma das cenas contidas no roteiro.

**[Calele, 2006] :** trabalho realizado em decorrência dos resultados obtidos em **[Calele, 2005].** O foco do estudo foi a geração de um Mapa de Intensidades Emocionais. Tal mapa foi criado para identificar a reação do usuário diante de Requisitos Não Funcionais (RNFs)

pretendidos pelo Game Designer 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.

**[Carvalho, 2006]:** o trabalho trata da criação de um processo preditivo de jogos para computadores pessoais. A grande inspiração para o modelo foi o RUP. O autor cria a característica de orientação pelo Game Design, o que significa que todas as atividades do processo devem ter um nível de detalhamento necessário no Documento de Game Design para que possam ser realizadas.

Além disso, há também o principio de gerenciamento do Documento de Game Design, que deve estar sempre alinhado as necessidades de entretenimento do jogador.

**[Alves, 2007] :** esse estudo realizado seguindo uma abordagem exploratória trata dos desafios de criação do Documento de Game Design e suas implicações no processo de Engenharia de Requisitos.

Durante todo o estudo, um Jogo foi utilizado e a analise de todo o desenvolvimento do mesmo gerou uma serie de dificuldades a ser discutida**,** as quais podemos destacar**:**

- *Melhoras no processo de criação do Documento de Game Design*

  Tal documento pode ser comparado a um documento de Requisitos, mas possui muitos elementos criativos como parte de seu conteúdo. O mesmo por sua vez não é tão padronizado quanto os documentos de Requisitos.

- *Mobile Games precisam atender a RNFs críticos*

  Os autores reconheceram que no processo de avaliação utilizado a compreensão sobre o quão satisfatório é o *gameplay* de um jogo ainda é um assunto em aberto, bem como a avaliação a respeito de outros RNFs críticos.

**[Furtado, 2009]:** o estudo foi realizado dentro do tema de fábricas de software a partir de uma abordagem baseada em desenvolvimento de domínio especifico. O processo iterativo criado para a produção do jogo é subdividido em quatro macro fases, cada uma delas trata de forma diferente a analise do Design do jogo e sua conseqüente implementação (última fase do processo).

**[Araujo, 2009]:** a proposta desse estudo, também orientado ao tema de fábricas de software, foi objetivada em aspectos de engenharia de aplicação (*application engineering*) para tratar de necessidades básicas à produção de jogos casuais. O autor faz uma extensa discussão a respeito dos métodos e métricas para salientar as decisões apresentadas em relação à modelagem do processo de desenvolvimento.

A forma de conduzir as informações do Documento de Game Design (DGD) para o software do jogo é obtida através de *estórias de usuários*, tomando-se os devidos cuidados para deixá-las fieis as características do DGD e suficientes em detalhes que permitam sua implementação.

# 2. O Documento de Game Design

Nos processos de desenvolvimento de Jogos não há nenhum documento que seja mais importante que o Documento de Game Design. Seu conteúdo tem a pretensão de apresentar todas as informações necessárias para o desenvolvimento do Jogo, 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 de um Jogo: Design, Artes, Tecnologia, Gerenciamento de Projetos, Psicologia 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 Elementos do Documento de Game Design.

A seguir definiremos alguns dos elementos que consideramos importantes para um Documento de Game Design. Tal definição ocorreu através do estudo realizado com base na literatura **[Scuytema, 2006] [Rollings, 2003][Schell, 2008]**, no processo de desenvolvimento de empresas de Jogos do Porto Digital **[Porto Digital, 2009]**
e de desenvolvedores do Centro de Informática da UFPE **[UFPE, 2009]**

Tais definição forneceram suporte para discussões com os grupos acima citados e para elaboração do *survey* da pesquisa.

- *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.

- *Personagens (controláveis ou não pelo jogador)*

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

- *Itens*

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

- *Design de níveis*

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

- *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.

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

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

- *Linha de arte*

O estilo de arte que definirá o visual do Jogo.

- *Telas de Jogo*

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

- *Trilha sonora*

As músicas que definem a sonoridade do Jogo.

- *Efeitos sonoros*

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

- *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.

- *Formas de pontuação*

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

- *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 Jogo.

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

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

- *Tema (idéia 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.

- *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.

# 3. Metodologias de Desenvolvimento de Software Aplicadas a Jogos

Essa seção tem o objetivo de apresentar as principais metodologias de desenvolvimento de software e sua utilização nos processos de desenvolvimento de jogos.

## 3.1 Papéis

Há uma grande variedade de papéis e atividades na produção de jogos eletrônicos. Abaixo listamos os mais significativos para este trabalho.

*Produtor* - O produtor é o responsável pelo planejamento e controle do Jogo. Possui uma visão macro do projeto e envolve-se com todas as fases do mesmo.. É responsável por garantir que o jogo seja desenvolvido conforme a especificação, o cronograma e o orçamento. É o papel mais próximo de um Gerente de Projetos entre todos os papeis de uma equipe de Jogos.

*Game Designer* - O papel do game designer é orientar, criar, desenvolver e documentar o fluxo do Jogo. É um papel que exige habilidades diversas como alto grau de abstração, criatividade, comunicação entre outras. Realiza atividades como: definir os elementos presente no jogo, a mecânica do fluxo do jogo, criar elementos centrais da história do jogo, personagens (controláveis ou não pelo jogador) etc.

*Artista* – O artista é responsável por definir os elementos artísticos propostos pelo Game Designer. Suas atividades compreendem a criação de artes conceituais dos personagens,*storyboards* que ilustrem o fluxo do Jogo, modelagem e animação 2D ou 3D, etc.

*Programador* - O programador é o responsável pela codificação do software que viabiliza a execução do jogo. Há várias especialidades de programadores dentro da equipe de desenvolvimento. Dependendo da especialidade do programador, este pode desenvolver o sistema gráfico, a inteligência artificial dos personagens, a infra-estrutura de comunicação, etc.

*Engenheiro de Áudio* - O engenheiro de áudio é o responsável por desenvolver os efeitos sonoros e a trilha musical do Jogo. Sua principal atividade é balancear tais elementos de acordo com a experiência de Jogo (gameplay) sugerida pelo Game Designer, etc.

*Engenheiro de Qualidade* - O engenheiro de qualidade é o responsável por medidas que levem o jogo a ter um nível de qualidade de acordo com os padrões exigidos pelo mercado. Desempenha atividades como planejar, monitorar e controlar testes de software, de arte, de jogabilidade, de usabilidade, etc.

## 3.2 Introdução

O inicio da produção de jogos eletrônicos seguiu uma tendência bem experimental, os jogos eram produzidos nas universidades, como o *SpaceWar*.

Nessa época, a produção de um jogo era uma tarefa que demandava uma quantidade excessiva de esforço técnico dado a falta de recursos para a programação de softwares. A implementação era primordialmente baseada em linguagens de máquina devido as arquiteturas disponíveis no momento, essencialmente utilizadas para problemas que demandavam mais performance de hardware e melhorias nas produções provenientes da engenharia eletrônica **[Ceruzzi, 2003]**

Embora pudessem fornecer aos engenheiros do período resultados de sucesso como o *SketchPad* **[Sutherland, 2003]**, o grande esforço envolvido para se criar um simples jogo, tornava essa tarefa demasiadamente custosa, principalmente por que envolvia um número bem reduzido de pessoas trabalhando , de uma forma primordialmente *ad hoc* e envolvia essencialmente uma única atividade: programação.

## 3.3 Saindo da Universidade

A partir do final dos anos 1960 e inicio dos anos 1970, a produção de jogos começou a tornar-se mais viável devido aos avanços das pesquisas em eletrônica e computação, que deram origens a linguagens com maior poder de abstração permitindo que os engenheiros passassem a pensar mais em seus softwares do que no esforço para programá-los.

Além disso, surgiu um mercado consumidor cada vez mais interessado em produtos tecnológicos. Para muitos a área de Entretenimento Digital floresceu durante esse momento.

Dois fatos importantes dessa época: a popularização dos *Arcades* e o posterior início da venda de consoles domésticos com a *Atari*, empresa criada em 1972 por Nolan Bushnell que com a colaboração de 7 estudantes tornou-se imediatamente um símbolo de uma geração e da historia dos videogames.

## 3.4 Rumo à maturidade

A partir dos anos 1970, da conseqüente evolução dos computadores e do surgimento de um mercado consumidor cada vez mais exigente os métodos *ad hoc* tornaram-se rapidamente insuficientes para dar conta da crescente demanda por títulos mais sofisticados em termos de vídeos, sons e desafios.

A indústria de jogos crescera de forma intensa e alcançar rapidamente a maturidade **[Araujo, 2009]** dentro desse mercado era uma necessidade mais do que evidente. Devido a tal situação, no decorrer dos anos subseqüentes, os processos de desenvolvimento de jogos passaram a adotar metodologias da Engenharia de Software, as quais veremos a seguir.

## 3.5 Processos baseados no modelo de desenvolvimento cascata (*waterfall*)

O modelo de desenvolvimento cascata está inserido dentro de uma **metodologia preditiva** para o desenvolvimento de software. A característica mais marcante desse

modelo é sua divisão em etapas bem definidas cujos resultados gerados passam de fase em fase seguindo uma única seqüência linear **[Sommerville, 2000].**
A adaptação de tal modelo para o processo de criação de jogos é conhecida como *Game Waterfall Process(GWP).* As etapas / fases de tal processo estão apresentadas na **Figura 1** abaixo.

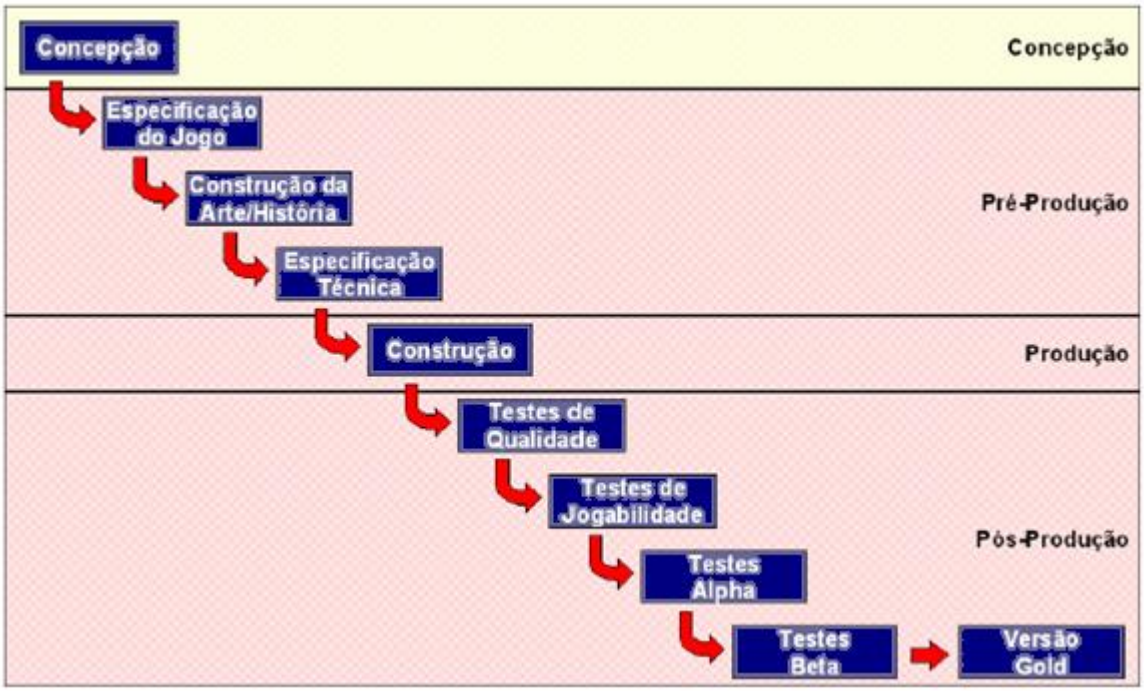



**Figura 1: etapas da criação de um jogo segundo o modelo cascata [Carvalho, 2006]**

Na prática, esse processo pode envolver mais ou menos etapas do que as apontadas, isso varia de acordo com a complexidade do jogo.

Apesar de possuir etapas bem definidas o GWP sofre várias críticas provenientes do modelo cascata da Engenharia de Software, pela sua forma estritamente seqüencial. Como uma fase do processo só pode ser iniciada após a conclusão de outra, acaba-se enfrentando o risco de se criar durante o ciclo único de concepção e desenvolvimento uma série de problemas de difícil solução, ocasionada pela natureza do processo.

Um exemplo clássico do tipo de problema encontrado no GWP diz respeito à fase de testes, como é iniciada após a implementação, caso seja verificado que a jogabilidade não está de acordo com o que os usuários esperavam para o jogo, a equipe terá um desafio de imensa proporção, visto a quantidade de retrabalho envolvida para descobrir em qual fase o problema foi originado e quais outras fases sofreram impacto do mesmo.

### 3.6 Processos baseados no *Rational Unified Process (RUP)*

O RUP é um processo de desenvolvimento criado pela *Rationa Softwarel* **[Rational, 2009]**, cujo principal objetivo é produzir softwares com alto nível de qualidade e que possa atender as necessidades dos seus usuários finais dentro de um escopo de requisitos e cronograma definidos com aprovação dos stakeholders envovlvidos **[Carrol, 2005]**.

.

Tal processo é organizado em ciclos, os quais trabalham em novas modificações de um mesmo produto. O ciclo é dividido em quatro fases consecutivas.

A conclusão de cada fase é definida por um *milestone* – um marco no desenvolvimento do projeto que leva em conta decisões criticas e reflexões sobre a subseqüente continuidade do projeto.

- *Concepção*
  - Durante a fase de **concepção** são estabelecidos regras de negócios e delimitação do escopo do projeto. Para isso são identificadas todas as entidades externas que interagem com o sistema, definindo a natureza dessa interação em alto-nível através de **casos de usos**.
  - Durante tal fase é possível a inclusão da Visão Geral do Jogo (High Concept) para analise por parte da equipe e *stakeholders* do projet

- *Elaboração*
  - O propósito da fase de elaboração é analisar o domínio do problema e estabelecer uma arquitetura do sistema, um plano de projeto e diminuir o acumulo de elementos de alto risco. Para isso é preciso que as decisões sejam tomadas tendo uma vista de toda a estrutura do sistema, suas principais funcionalidades, o escopo e o tempo necessário de produção.
  - Tal elaboração pode se estender ao planejamento da produção dos elementos artísticos do Jogo gerando uma estimativa total de projeto mais precisa.

- *Construção*
  - Durante a fase de construção, todos os componentes e características da aplicação são desenvolvidos e integrados ao produto. É sobretudo uma fase de manufatura, onde a ênfase é dada no gerenciamento de recursos e controle operacional para otimizarão de custos, cronograma e qualidade.
  - Nesta etapa podem ser incluídos os testes de usabilidade e as avaliações de interface objetivando a descoberta de funcionalidades que possa ser melhoradas e aperfeiçoadas para o lançamento do produto.

- *Transição*
  - O propósito da fase de transição é transferir o produto para sua comunidade de usuários finais, uma vez que o produto está de posse de seus usuários, frequentemente são requisitados novos desenvolvidos e lançamentos corrigindo alguns problemas finalizando o conjunto de características definido.
  - Em tal etapa podem ser utilizadas pesquisas de satisfação para compreender o impacto do Jogo entre seus usuários com base em emoção e valor e identificar se vale ou não continuar investido no lançamento do Jogo.

Essas quatro fases são orientadas por dois grupos de disciplinas: engenharia e suporte.

*As disciplinas de engenharia são:*

*Modelagem de negócios* – Na modelagem de negócios são documentados processos comumente chamados de casos de uso de negócios. Nada mais é do que uma forma de garantir um comum entendimento entre todos os *stakeholders* das necessidades da organização para suporta o processo em desenvolvimento.

*Requisitos* – A meta dos requisitos é descrever o que o sistema poderá fazer e permitir aos desenvolvedores e outros envolvidos na produção o comum acordo entre essas funcionalidades. Para isso, são elicitados, organizados e documentados de acordo com sua requerida funcionalidade.

*Analise & Design* – A meta da fase de Analise & Design é especificar com clareza como será realizada a implementação do sistema. O resultado é um modelo que serve como uma abstração do futuro código fonte.

*Implementação* – O propósito dessa fase é definir a organização do código em termos de subsistemas definidos em camadas para implementar classes e objetos em termos de componentes (códigos fontes, executáveis e outros). Além disso, os códigos produzidos são testados e integrados para formar um único sistema.

*Testes* – O propósito dos testes é conseguir verificar a interação entre os componentes do software e identificar se foram corretamente implementados, quais as falhas apresentadas e as prioridades para solucionar tais problemas.

*Implantação* – A implantação objetiva o lançamento e a entrega do produto a seus usuários finais. Cobre um grande numero de atividades, tais como distribuição e instalação

*As disciplinas de suporte são:*

*Gerenciamento de projetos* – disciplina encarregada de equilibrar os objetivos do projeto, os riscos envolvidos, as diferentes atividades, os prazos e os interesses dos *stakeholders* em relação ao produto final.

*Gerenciamento de configuração & mudanças* – descreve como controlar os numerosos artefatos produzidos pela equipe e garantir a conformidade dos mesmos com o objetivo de evitar dúvidas e conflitos em relação a problemas como múltiplas versões e atualizações simultâneas.

*Ambiente* – O propósito da ambientação é prover o ambiente de desenvolvimento de software de forma adequada com as ferramentas e procedimentos necessários para a equipe de trabalho.

O RUP foi adaptado para o desenvolvimento de jogos pelo Engenheiro de Software Kevin Flood, que após ter enfrentado sérios problemas com o desenvolvimento de um título passou a usar um processo semelhante, entretanto orientado também a criação de conteúdo. A adaptação recebeu o nome de *Game Unified Process (GUP)* **[GameDev, 2009]**.

O conhecimento das etapas que compõem o RUP e do funcionamento do processo como um todo é essencial para a criação de modelos baseados no mesmo **[Carvalho, 2006] [GameDev, 2009]**

Basicamente, o importante a se considerar numa adaptação de tal processo é saber em quais etapas incluir os aspectos de produção artística e de Design do Jogo. Ou então, se será preciso criar etapas separadas para essas tarefas. As fases do RUP, bem como uma descrição a respeito do que ocorre em cada uma delas e quais podem permitir a inclusão de elementos de Design e Artes estão listadas abaixo.

### 3.7 Processos orientados à Metodologias Ágeis

As metodologias ágeis foram adotadas em reação aos modelos de desenvolvimento excessivamente burocráticos e restritos como o GWP e o RUP por exemplo.
Basicamente, são alternativas encontradas ao longo dos anos pelos desenvolvedores para se obter um paradigma de produção mais flexível a mudanças, porem com o mesmo nível de controle das metodologias previamente apresentadas.

A popularização das metodologias ágeis ocorreu a partir de 2001 quando um renomado grupo de engenheiros de software publicou o manifesto ágil, no qual apresentaram uma serie de valores e princípios essenciais ao desenvolvimento de projetos, mas que estavam em desuso pelos métodos mais tradicionais.

Essas metodologias são recomendadas para projetos com times pequenos e que possuam a característica de mudanças freqüentes em seus requisitos. O que implica em uma necessidade de adaptação e *re-design* constante, pouco comum em outras metodologias.

Entretanto já é possível observar o uso de tais metodologias em projetos mais longos e com times bem maiores na própria área de Jogos **[Agile Game Development, 2009]**.

Para isso as metodologias possuem focos diferenciados das metodologias já citadas. A produção do software é mais intensa do que o trabalho de documentação, a comunicação entre desenvolvedores, clientes e usuários é constantemente incentivada, bem como a elaboração de planos alternativos decorrentes das mudanças (do projeto, de membros da equipe, etc.) ocorridas durante o desenvolvimento.

### 3.8 Extreme Programming XP

Assim como nas demais metodologias ágeis, o XP é uma metodologia de desenvolvimento de software que se diferencia por valorizar a adaptação à mudanças mais do que a capacidade de pré-ditar o desenvolvimento, por conseqüência da constante mudança de requisitos observada pelos avaliadores do XP.

Dessa forma, o XP é recomendado para times de desenvolvedores que possuam as seguintes características:

- O time é pequeno;
- O time trabalha com requisitos de difícil interpretação
- O time trabalha com constantes mudanças;
- O time tem a preocupação de fornecer garantias aos clientes.

O desenvolvimento de um software XP é radicalmente diferente das outras metodologias. Não há seqüências de fases com séries de artefatos sendo produzidos ao longo da implementação. O que em relação ao desenvolvimento de Jogos acaba por apresentar certa incoerência já que classicamente a produção de jogo acontece em torno de um artefato, o Documento de Game Design **[Schuytema, 2006]**.
No XP, três elementos são de essencial importância para desenvolver o software: as *estórias*, os *releases* e as *iterações*.

*Estórias* – São descrições em alto nível de necessidades do usuário/cliente narradas pelo próprio. As **estórias** são avaliadas pelos membros do time de forma a se obter um conjunto formal de funcionalidades para o sistema, cujas prioridades são definidas em comum acordo entre clientes/usuário e equipe.

*Releases* – Os **releases** são módulos do sistema que compreendem um subconjunto das **estórias**, as quais possuem fortes valores de negócios e são as principais fontes para que a equipe de desenvolvimento implemente o sistema.

*Iterações* – As iterações são períodos curtos de desenvolvimento em que as funcionalidades do sistema são implementadas. A prioridade das funcionalidades é associada a importância atribuída as estórias pelos clientes e geralmente seu tempo médio é de duas semanas.

### 3.9 Valores do XP

A metodologia de desenvolvimento XP é orientada a valores que guiam a equipe por todo o ciclo de desenvolvimento do projeto. Tais valores prezam pela harmonia do trabalho em equipe e também pelo respeito aos valores individuais de cada membro, o que acaba sendo vantajoso para o desenvolvimento de Jogos devido a multidisciplinaridade das equipes **[Araujo, 2009]**.

*Simplicidade* - significa fazer apenas o que é preciso e nada mais. Isso maximize o esforço da equipe em resultados respeitando o investimento do projeto. Assim, adotam-se planejamentos de pequenos passos com metas precisas que facilitam a resolução de falhas assim que elas ocorrem O principal objetivo é criar algo que seja digno de nota, de orgulho para equipe honrando a confiança dos clientes.

*Comunicação* - todos são partes de um mesmo time e a comunicação cara-a-cara é sempre encorajada diariamente. Todo o trabalho é feito conjuntamente, da coleta de requisites a codificação. A crença de que problemas são solucionados de melhor forma em companhia do que solitariamente é seguida a risca.

*Feedback* - toda iteração será encarada com muito comprometimento pela equipe sempre objetivando entregar um software executável ao final. Assim, o software será avaliado pelos *stakeholders* e suas impressões cuidadosamente verificadas para modificações e correções no projeto. Dessa forma, o trabalho se adapta as necessidades do cliente de forma mais segura e fácil.

*Respeito* - Todos no time terão o mesmo devido respeito**.** A contribuição de toda a equipe

garante entusiasmo e boas práticas de desenvolvimento. Desenvolvedores respeitam a experiência dos clientes, bem como o pessoal da gerência respeita as opiniões sobre seu próprio trabalho da mesma forma que são respeitados pela responsabilidade de suas funções.

*Coragem* **-** Ter sempre a coragem de comentar sobre progressos e estimativas. O objetivo é manter o time unido impedido que qualquer um sinta-se sozinho e temeroso sobre o andamento de suas atividades, permitindo ao time, adaptação rápida a mudanças no momento em que elas ocorrem.

A metodologia XP respeita a individualidade de cada time e trata os valores dos mesmos com muita propriedade. A adição de novos valores que façam sentido para uma equipe ou um novo projeto é sempre incentivada. Isso faz com que os membros do time adquiram capacidades de refletir sobre suas próprias atividades, fornecendo experiências que farão com que os mesmos venham a tornar-se cada vez mais capacitados.

## 3.10 Práticas do XP

As práticas são um subconjunto de atividades executadas diariamente dentro do ciclo de desenvolvimento de *software*.

São definidas regras claras e concretas para as praticas sempre com o compromisso de satisfazer valores importantes para a equipe e o trabalho. Funciona também como uma forma de integrar a equipe, tornando – a mais suscetível a mudanças de contexto.

As práticas da XP são úteis tanto isoladamente quanto em conjunto, embora seja mais eficiente quando aplicada em grupo por conseqüência dos vários valores reunidos o que implica na melhoria dos resultados.

O próprio manifesto apresenta um conjunto de práticas a serem seguidos, mas de forma alguma faz exigências sobre qual prática tem de ser mais ou menos usada. Muito pelo contrário, a metodologia XP incentiva a própria equipe a descobrir (ou até mais) quais são as melhores práticas dentro do contexto em que a empresa está interessada e do projeto a ser desenvolvido.

São dois os tipos de praticas reconhecidos pelo XP: Primárias e Corolárias, as primeiras dizem respeito a provocar melhorias no andamento dos projetos, enquanto que as segundas são dedicadas a dar continuidade aos benefícios, desde que as práticas Primárias tenham sido bem executadas.

Abaixo uma pequena lista de alguma das Práticas desenvolvidas pelo XP.
*Cliente Presente* – A presença do cliente é fundamental para que os envolvidos no projeto tenham o parâmetro correto para entender quais as verdadeiras necessidades do mesmo – e o que é preciso fazer para atende-las da melhor forma possivel.

*Stand Up Meeting* - Reuniões de projeto na qual os membros da equipe se encontram de pé para comentar rapidamente sobre o desenvolvimento do projeto.

*Refactoring* - O código deve ser constantemente revisto para que seja suficientemente simples.

*Código Coletivo* - Todos os desenvolvedores podem se sentir à vontade para modificar qualquer parte do código que seja do seu interesse.

*Ritmo Sustentável* – Toda a equipe deve manter um ritmo sustentável de desenvolvimento sem que haja a necessidade de horas extras.

.

*Integração Contínua* – Tem o objetivo de assegurar que todos os módulos que compõem o sistema estejam sempre funcionando.

*Releases Curtos* – Trabalhar com pequenos prazos de release de forma a gerar um fluxo contínuo de entregas a serem avaliadas pelos clientes.

### 3.11 Extreme Game Development (XGD)

O XGD é uma adaptação do XP à criação de jogos digitais. Bem como na área de desenvolvimento de softwares, o XGD foi adotado por ser uma metodologia mais orientada a apresentar resultados em prazos curtos e bem definidos, com avaliação constante dos clientes devido a adoção das prática anteriormente citadas.

A adoção do XGD aconteceu também pela necessidade maior de controle que as empresas necessitavam para sua produção, as metodologias não ágeis sempre colocavam a equipe de desenvolvimento em situação de risco devido à dificuldade de controlar prazos e as conseqüentes falhas de sistema provocadas pela sobrecarga de atividades.

Apesar dos valores, das práticas e dos princípios o XGD não é suficiente para tratar da produção de um jogo. As maiores criticas feitas **[Carvalho, 2006]** são provenientes da adaptação ter surgido originalmente da Engenharia de Software. Como criar um jogo não é apenas criar um software, muitas disciplinas de igual importância como o Design e a Arte não recebem benefícios da metodologia XP.

Além disso, é característica da área de jogos que a produção dos mesmos ocorra em torno de um documento central (DGD). O que gera um conflito com o XP, já que é de sua natureza a quase ausência da geração de documentos.

Elementos importantes do XP como as práticas e os valores também se mostram incompletos quando observados pela ótica da produção de Jogos, pois apesar de serem interessantes por demonstrarem união e valorosos ganhos de experiência por parte da equipe, são fortemente associados a atividades de software, o que implica sempre em revisões para perceber quais suas melhores adequações ao desenvolvimento de jogos.

# 4. Metodologia de Trabalho

A metodologia do trabalho foi dividida nos seguintes aspectos.

## 4.1 Estudo Bibliográfico

- Etapa na qual foram levantados fontes de pesquisa que continham estudos semelhantes relacionados ao desenvolvimento de jogos nas áreas de Engenharia de Software e Metodologias de Design.

- O resultado desse estudo bibliográfico foi direcionar os esforços deste trabalho a uma etapa do processo de criação de jogos e não na criação de um novo processo.

- Tal escolha justifica-se pela quantidade de processos de Engenharia de Software atuando ou inspirando a criação de novos processos para o tema. Sendo assim, procuramos gerar uma alternativa a um problema comum entre os projetos verificados: o conflito entre o DDG e a implementação do jogo.

## 4.2 Entrevistas Semi-estruturadas

- Após a etapa inicial definimos um pequeno conjunto de questões para uma série de entrevistas informais semi-estruturada, essas entrevistas foram respondidas por produtores, engenheiros e designers nos seus próprios locais de trabalho. O objetivo foi de extrair opiniões a respeito de como eles consideram o que poderiam ser boas alternativas para resolver os conflitos de *DGD Vs. implementação do jogo.*

- O tempo desse Estudo de Caso durou aproximadamente três semanas, foi realizado em três empresas do Porto Digital **[Porto Digital, 2009]** e através de um grupo de desenvolvimento de jogos da UFPE **[UFPE, 2009]**.

- O resultado foi a geração de um *survey* baseado inteiramente no conhecimento adquirido das duas fases até então.

## *4.3 Survey*

- O *survey* foi gerado para obter um número maior de participantes da pesquisa. Dessa forma conseguimos um bom resultado da visão que funcionários de empresas nacionais ou estrangeiras, além de desenvolvedores independentes ou associados a departamentos acadêmicos.

- A partir dos três itens de dados (estudo bibliográfico, entrevistas semi-estruturadas e *survey*) sobre a produção de jogos e todo o acúmulo de informações decidiu-se que já se havia um conhecimento necessário para iniciar

uma análise mais apurada acerca de todos os dados obtidos tendo o *survey* como principal fonte dos dados.

## 4.4 Caracterização dos Participantes

Contamos com a participação de 38 pessoas, todas envolvidas de alguma forma com a produção de jogos eletrônicos, seja no meio acadêmico ou industrial.

### *4.4.1 Atividades dos Participantes*

A grande maioria dos participantes **[Tabela 1]** concentram-se em atividades relacionadas ao Design (26%) e a Programação de jogos (34%), dado que pode ser ainda maior ser incluímos os Engenheiros de Software (8%) nesta mesma categoria pela proximidade da relação que envolve as duas atividades.

**Tabela 1: percentual dos pápeis na indústria de jogos (amostra global = 38 participantes)**

*Qual o seu papel na Indústria de Jogos*

| | | |
|---|---|---|
| Game Designer | 10 | 26% |
| Produtor | 5 | 13% |
| Programador | 13 | 34% |
| Artista | 4 | 11% |
| Engenheiro de Software | 3 | 8% |
| Outros | 3 | 8% |

### *4.4.2 Experiência dos Participantes*

Quanto à experiência **[Gráfico 1],** a grande maioria dos participantes possui entre 3 e 5 anos de experiência (32%) ou menos de um ano (29%). Praticamente, metades desse percentual estão os membros mais experientes da nossa pesquisa, com mais de 5 anos de experiência na nossa indústria de jogos (13%).

### *4.4.3 Origem dos participantes*

Os participantes do *survey* são profissionais da Indústria de Jogos em Pernambuco ou estudantes de instituições de ensino superior no mesmo estado. Boa parte deles acessaram a pesquisa através do capitulo estadual da International Game Developers Association **[IGDA, 2009]**.

Boa parte dos participantes forneceram dados pessoais, os quais permitiram que criássemos três amostras de suporte para a nossa pesquisa.

- Indústria Nacional – 13 participantes
- Indústria Internacional – 4 participantes
- Universidade – 8 participantes

Seus resultados serão citados sempre que revelarem dados importantes para informações conclusivas a respeito do resultado da amostra global.

**Gráfico 1: experiência dos participantes**


### *4.4.4 Métodos de Análise*

Para nossa analise de dados, foi usada uma metodologia mista: baseada tanto em métodos qualitativos quanto quantitativos.
Tal principio de análise vem sendo bem aceito na geração de alternativas para produção de softwares orientados a criação de conteúdo educacional **[Sedig, 2001]**, conteúdo artístico para modelos arquitetônicos **[Young Oh, 2006]** ou industria de entretenimento **[Shin, 2007]**.

#### *4.4.4.1 Análise Qualitativa*

A analise qualitativa foi gerada inteiramente a partir das opiniões fornecidas pelos participantes do *survey*.

Utilizamos o auxílio do software nVivo**[NVIVO, 2009]**, destinado a facilitar trabalhos de pesquisa puramente qualitativa. Optamos por não seguir nenhuma técnica de codificação abordada na literatura **[Flick, 2004]**, mas sim um conjunto de características apresentadas na codificação axial e codificação seletiva como a descoberta de padrões entre as categorias criadas de forma livre, o enriquecimento através de citações nas transcrições das entrevistas e dos comentários presentes no *survey*, além da criação de possíveis relações. Tal decisão se apresentou mais fiel aos propósitos da pesquisa.

#### *4.4.4.2 Analise Quantitativa*

A análise quantitativa foi feita com base em toda a amostra e com os dados auxiliares das amostras por grupos (industria nacional, industria internacional, universidade). Em todos os casos foram geradas distribuição de freqüência e visualização através de gráficos de barra ou em gráficos de *pizza*. Todos os cálculos foram gerados através da ferramenta *survey resume* do Google Documents **[Google Documents, 2009]** ou do *National Center for Educational Statistics – NCES* [**NCES, 2009]**.

# 5. Resultados

Os resultados são apresentados sob a forma de um conjunto de *guidelines* **[Young Oh, 2006][Sommerville, 1997]** e originado com o foco em todo o conhecimento obtido a partir dos processos de desenvolvimento de jogos vistos nesse trabalho e dos itens descritos no capítulo anterior.

## 5.1 *Guidelines* e Game Design

*Guidelines* são muito usados na indústria de jogos para resolver problemas de Design, sobretudo no que tange a Experiência do Usuário em relação ao *Gameplay* concebido.

Um exemplo da eficiência do uso de *Guidelines* no design de jogos é o estudo desenvolvido para orientar designers na concepção de jogos para Realidade Aumentada (RA) **[Wetzell, 2008]**.

Todo o estudo foi trabalhado com foco em exemplos reais da industria, como o jogo *The Eye Of Judgemt* **[The Eye of Judgement, 2009]** para o console Playstation 3 **[PlayStation, 2009]**.

As conclusões revelaram uma série de *Guidelines* sobre como desenvolver uma boa experiência de jogo que possa se adequar as expectativas dos jogadores observando as limitações e oportunidades do paradigma de RA.

Um outro projeto – *Game Over!* **[Grammenos, 2008]**, trata da criação de *guidelines* para orientar o desenvolvimento de games cujo requisito mais importante é a acessibilidade. Para atingir tal meta, o autor desenvolveu o "*Game Over!",* um jogo que segundo o próprio autor não pode ser jogado por ninguém.

O objetivo principal foi o de obter dados a partir dos jogadores sobre como devem ser implementados jogos acessíveis – possíveis de ser jogados por qualquer pessoa.
O autor chegou a gerar em torno de 20 *guidelines*, todos eles devidamente definidos segundo a necessidade dos usuários participantes, dessa forma haviam *guidelines* para melhorar detalhes do jogo para jogadores com deficiência motora, visual, auditiva, não deficientes e até mesmo para todos os esses casos ao mesmo tempo.

## 5.2 *Guidelines* e Engenharia de Software

O uso de *guidelines* na Engenharia de Software também é muito extenso e atua em praticamente todas as áreas da disciplina.

Nesse trabalho **[Ramachadran, 2005]** o autor desenvolve uma serie de categorias para uso de *guidelines* orientados ao reuso de software. Na pesquisa, a tentativa foi de tornar mais próximo o conhecimento do domínio de uma aplicação do conhecimento de domínio da linguagem que representará tal aplicação. A idéia foi usar o conjunto de *guidelines* desenvolvidos para dar suporte a analises sobre o reuso de estruturas de dados tornando todo o domínio de conhecimento mais compreensível e produtivo.

No livro *Requirements Enginnering – A good practice guide* **[Sommerville, 1997]**, o autor descreve detalhadamente uma serie de mais de 60 *guidelines*. A obra, apesar de manter o foco na Engenharia de Requisitos, acaba por apresentar orientações que permeiam todo o ciclo de desenvolvimento de software. A riqueza de comentários e exemplos

disponíveis para os *guidelines* mostram a clareza e o bom impacto que o uso dessa técnica ocasiona na indústria de desenvolvimento de software.

## *5.3* Apresentação dos *Guidelines*

Os *guidelines* serão apresentados de acordo com uma adaptação da estrutura apresentada por **[Sommerville, 1997]**, na qual cada *guideline* aparece seguido de uma lista de seus principais **benefícios** e uma estimativa do(s) **custo(s)** para sua implantação dentro dos processos**.** A maior parte dos *guidelines* desse estudo foram obtidos através dos dados quantitativos e qualitativos da pesquisa, entretanto alguns foram inteiramente baseados na lista de *guidelines* disponnivel em **[Sommerville, 1997]**, pela forte relação com os propósitos deste trabalho e a similar necessidade da aplicação dos mesmos ao contexto da produção local de jogos.

### *5.3.1 Estimativa dos Custos*

Estimamos os custos dentro de três categorias: **Baixo**, **Moderado** e **Alto**. Os custos foram apontados segundo as definições encontradas em [requirement analyses] e [baxter] por estarmos tratando de uma etapa de processo que envolve levantamento de requisitos e especificações detalhadas para produção. Além disso, muitos dados qualitativos também foram usados para tal.

### *5.3.2 Disney Creativity Strategies*

Devido aos processos vivenciados pela maioria dos participantes do trabalho, percebemos que para muitos, alguns detalhes apresentados para melhorar o desenvolvimento dos Jogos ainda estão em um estágio considerado utópico e de pouca possibilidade de aplicação prática.

Dessa forma decidimos apresentar os *guidelines* de acordo com uma adaptação do *Disney Creativity Strategies* **[Mycoted, 2009].** Trata-se de uma técnica criativa para concepção de produtos criada por Robert Dillts a partir da análise que o mesmo fez dos processos criativos utilizados nos estúdios Disney **[Disney, 2009]**, bem sucedido em transformar sonhos utópicos em realidade segundo o próprio Dillts.

Em tal técnica, os projetos têm três fases de concepção:

- **Sonho** – São definidos sem nenhuma preocupação. A imaginação é o limite.
- **Realista –** Aqui o projeto imaginado no **sonho**, passa por avaliações para se determinar o que pode ser realizado.
- **Critico** – Na etapa final, a avaliação **realista** passa por um plano de testes, que identifica os principais problemas e viabilidades da realização do projeto**.**

Como já iniciamos de um modelo de processo pré-definido, apresentamos os *guidelines* pelos seus ideais **Realistas** e ideais **Críticos.**

**#1** **Game Designer presente em todo o processo**

***Benefícios***

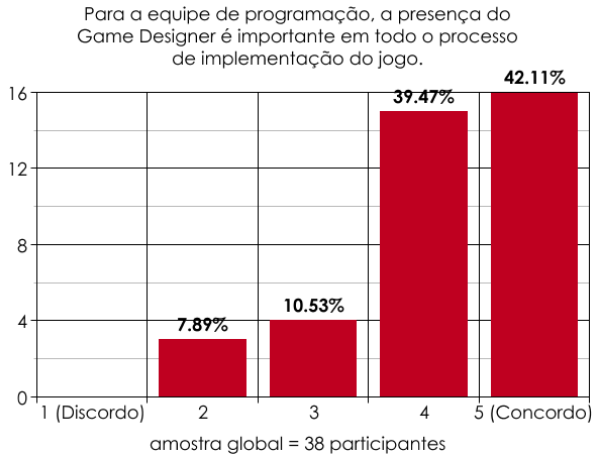
*•Redução do esforço em compreender informações do DGD.*
*•Maior facilidade para encarar mudanças em determinadas características do jogo.*
*•Determinação das etapas do processo nas quais a presença do Game Designer é fundamental*

***Alto Custo***

*•Dificuldades em garantir a prontidão do Game Designer em todo o processo.*

Um dos fatores mais recorrentes tanto na fase das entrevistas quanto nas respostas ao *survey* foi quanto à presença do *Game Designer* durante as fases que compõem o processo de desenvolvimento. No **[Gráfico 2]** é possível identificar a importância atribuída a tal presença para a implementação do Jogo.

**Gráfico 2: importância do Game Designer no processo de implementação**




## Ideal Realista

Com base na análise quantitativa, podemos afirmar que aproximadamente 82 % (junção dos níveis de concordância 4 e 5) dos participantes concordam que a presença do Game Designer tem fundamental importância em todo o processo de implementação do jogo. Como o percentual da maioria das respostas nas **amostras por grupo [Gráficos 3, 4 e 5]** correspondeu a da **amostra geral** não houve dúvidas em classificar essas informações como um ideal realista.

## Ideal Crítico

Entretanto, há casos nos quais a presença do Game Designer não é possível no momento em que surgem as dúvidas dos Programadores (ou mesmo dos artistas). As causas mais possíveis para isso é o envolvimento do Game Designer em vários projetos, reunido

com *stakeholders* ou mesmo tratando de detalhes técnicos e artísticos com os lideres dessas duas áreas, bem como discutindo prazos para entrega dos resultados de algum dos testes ou da entrega final do Jogo.

Qualitativamente observamos que a presença do Game Designer, basicamente é necessária principalmente em atividades nas quais o jogo está sendo avaliado de alguma forma (**comentários 1, 2 e 3**).

***"Sim, existem momentos mais críticos...***
***...elaboração dos protótipos (denifinição dos limites dos protótipos e o que se deseja atingir com eles)" [comentário 1]***

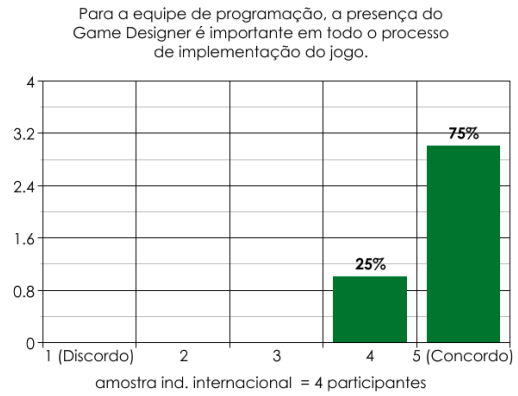
***"... para acompanhar progressivamente os resultados de implementação de cada milestone" [comentário 2]***

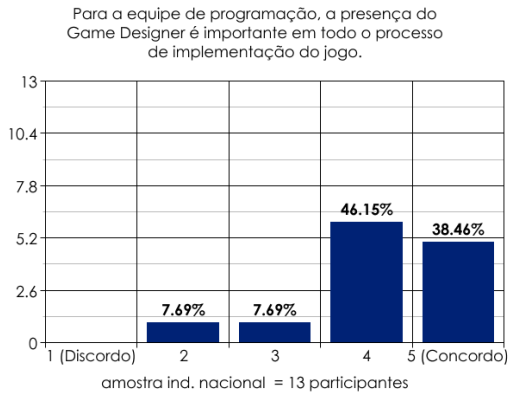
***"No inicio (planejamento) e no final (ajustes finos, balanceamento)" [comentário 3]***

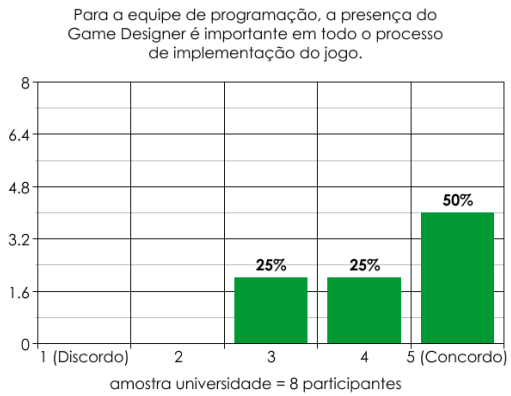
Para evitar que haja uma grande quantidade de tempo sem resposta, uma alternativa é listar momentos críticos do desenvolvimento onde a presença do Game Designer é fundamental, dessa forma o mesmo concentra-se na definição de um cronograma (possivelmente com auxilio do Produtor) no qual sua disponibilidade para a equipe seja maior em fases nas quais a avaliação do jogo esteja ocorrendo.

- Fases de Prototipagem;
- Fases de Testes de Jogabilidade;
- Fases de Testes de Usabilidade;
- Fases de Balanceamento do Jogo;

**Gráficos 3, 4, 5: importância do Game Designer no processo de implementação para as diferentes amostras da pesquisa (esquerda(3) - indústria internacional, direita(4) - indústria nacional e abaixo(5) - universidade)**



**#2** **Criar Visão Geral (High Concept) Colaborativa**

***Benefícios***
*•Maiores garantias de compreensão do projeto por parte da equipe;*
*•Reduz tempo de escrita e esforço do Game Designer em comunicar a mesma idéia para toda equipe.*

***Alto Custo -*** *dificuldades em conseguir uma colaboração massiva dos envolvidos no projeto.*
***Alto Custo*** *- Esforço considerável em conceber uma Visão Geral multidisciplinar e colaborativa.*

Dos elementos que compõem um DGD, a visão geral (*high concept*) do jogo foi considerado o mais importante nas fases preliminares do desenvolvimento do game. Aproximadamente 97% dos participantes do *survey* **[Tabela 2]** definiram a visão geral como o item mais significativo para se iniciar o desenvolvimento de um game a partir de um DGD preliminar.

**Tabela 2: destaque para a Visão Geral (High Concept) do Jogo como item mais importante para o inicio do processo de desenvolvimento**

| | |
|---|---|
| Visão Geral (High Concept) do jogo | 97% |
| Personagens (controláveis ou não pelo jogador) | 55% |
| Itens | 30% |
| Design de níveis | 30% |
| Fluxo do game | 82% |
| Controles (ações do jogador com o personagem) | 85% |
| Linha de arte | 30% |
| Telas de jogo | 30% |
| Trilha sonora | 3% |
| Efeitos sonoros | 6% |
| Regras do jogo | 88% |
| Formas de pontuação | 61% |
| Análise de competidores | 21% |
| Referências (filmes, jogos, livros, etc.) | 48% |
| Tema (idéia filosófica) | 27% |
| Estilo (ação, estratégia, puzzle) | 73% |
| Outros | 12% |

**amostra global = 38 participantes**

## Ideal Realista

O ideal é que a equipe possa participar da construção dessa visão, tal colaboração é orientada a criar uma visão do projeto que seja sensível as várias disciplinas do desenvolvimento de jogos.

Dessa forma, Designers, Programadores, Artistas, Produtores e outros *stakeholders* possibilitam ao Game Designer conceber uma Visão Geral mais definida aos propósitos apontados pela equipe, com maior possibilidade de evitar futuros conflitos de compreensão (**comentário 4**).

> ***"... Se essa visão não fica clara, problemas certamente surgirão mais a frente..." [comentário 4]***

A tendência é que tal definição possa economizar tempo de interpretação do time e tempo de escrita e esforço do Game Designer em saber como comunicar uma mesma idéia para pessoas que desempenham tarefas bem distintas umas das outras, ainda que complementares.

## Ideal Crítico

Inúmeros fatores podem impedir que haja uma colaboração massiva para construção única de uma Visão Geral do Jogo. Para tal situação é recomendado que a Visão Geral a ser criada pelo Game Designer possa contar com pelo menos a participação de um representante da equipe de Programadores e um representante da equipe de artes, que podem ser os respectivos Líder Técnico e Líder Artístico.

Caso, ainda assim seja difícil contar com a disponibilidade de outros membros da equipe, o Game Designer deve construir a Visão Geral pensando na heterogeneidade do time (**comentário 5**).

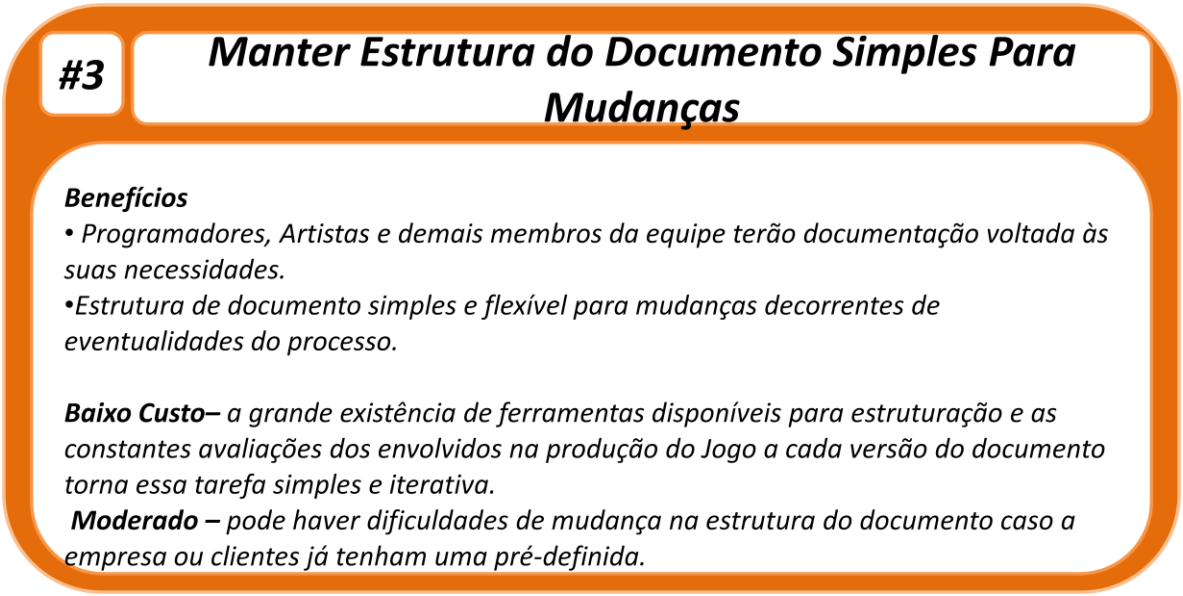
> ***"o Game Designer (ou os responsáveis pelo Game Design) deve atuar proativamente***
>
> ***para garantir que a visão geral do conceito torne-se cada vez mais clara e uniforme na cabeça de todos os envolvidos no projeto" [comentário 5]***

Essa visão geral colaborativa tem inspiração em alguns valores do XP, sobretudo:

*Simplicidade* – criar uma visão geral de forma colaborativa permita a todo time de desenvolvimento a se habituar cedo com a importância de fazer apenas aquilo que deve ser feito.

*Respeito* – a colaboração sendo incentivada numa concepção conjunta faz com que todos, sem exceção, possam compreender da mesma forma os propósitos do projeto.

*Coragem* – a colaboração nessa etapa ou em atividades semelhantes ajuda a criar no time um espírito mais questionador e seguro para discutir eventuais falhas da atividade ou de qualquer outra etapa futura do processo de desenvolvimento.




Como o DGD após o lançamento de sua primeira versão estará (ou até mesmo antes) sujeito a constantes revisões e atualizações é importante que o Game Designer possa definir uma estrutura do documento que seja simples e eficiente nas tarefas de re-edição e re-distribuição para os membros das equipes **[Sommerville, 1997]**.

**Ideal Realista**

O interessante é que o Designer possa definir essa estrutura de forma orientada a cada grande área envolvida na produção do jogo. 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 documentos 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 requisitar que a seção técnica do documento seja definida como uma lista de requisitos contendo detalhes como a prioridade do mesmo, os atores (jogadores ou sistemas) que poderão executar tal requisito e os conteúdos de arte associados (caso existam). Dessa forma, Programadores e Game Designer estarão alinhados (**comentário 3**) com as mesmas idéias sobre o desenvolvimento do Jogo, como ratificado pelo comentário abaixo.

***"Sempre deve haver um alinhamento entre a equipe de programação (ou ao menos o Lead) e o Game Designer" [comentário 3]***

Ou então, os Programadores podem decidir por uma estrutura baseada em estórias associadas a casos de uso com prioridades definidas de acordo com o nível de esforço de implementação avaliado pela equipe **[Moløkken-Østvold, 2008]**.

## Ideal Crítico

Independentemente da escolha definida pelos Programadores e designers ou da definição prévia da instituição sobre o modelo do documento, este deverá obrigatoriamente estar apto a receber dúvidas, criticas e sugestões.

O interessante é que isso possa ocorrer diretamente no seu conteúdo para que seja possível a discussão dos problemas de forma igualitária por ambos os pontos de vista até que se encontre um meio – termo suficiente para a implementação de alguma funcionalidade do jogo permitindo a sintonia entre o Game Designer e os Programadores da equipe (**comentário 4)**.

***"...antes da implementação sair do papel, é essencial que a equipe e o designer estejam sintonizados nos mínimos detalhes."[comentário 4]***

## #4 Criar Canal de Comunicação Direta com o Game Designer

***Benefícios***

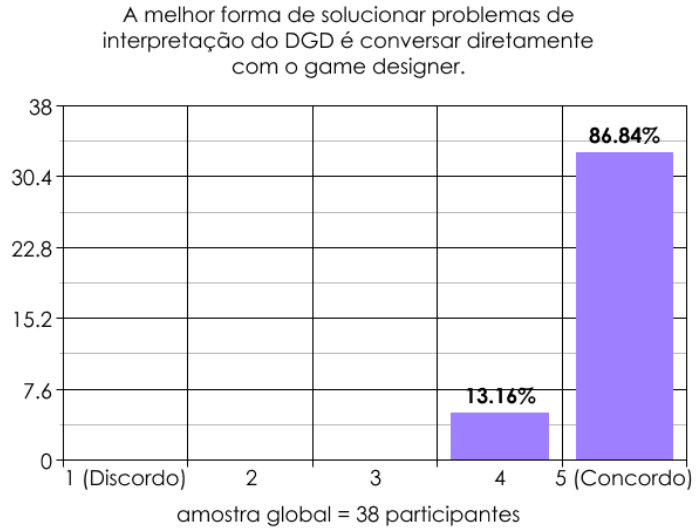
*•facilita comunicação da equipe com o Game Designer;*
*•Organiza as dúvidas da equipe por prioridade;*
*•Ao final de cada projeto, uma boa quantidade de informações estará disponível para orientar trabalhos futuros.*

***Alto Custo-*** *implementar tal mecanismo implica em outros custos para a empresa como a necessidade de constante manutenção e gerenciamento.*
***Custo Moderado –*** *o uso de ferramentas gratuitas é uma boa alternativa. Mas nem sempre permitem a possibilidade de se adequar a todas as necessidades da equipe.*

Apesar de todos os detalhes contidos em um DGD e de todo o cuidado nas subseqüentes revisões do texto que o compõe. É comum que em muitas situações a equipe de programação tenha dúvidas sobre a implementação de alguma funcionalidade especifica do jogo, o que é uma conseqüência óbvia do dinamismo envolvido no processo de desenvolvimento.

Os participantes do *survey* em sua grande maioria **[Gráfico 6]** indicaram que a melhor forma de sancionar essas dúvidas é se dirigindo diretamente ao Game Designer.

**Gráfico 6: níveis de concordância quanta a solução para problemas de interpretação do DGD**




Entretanto, mesmo estando coordenando uma equipe em todo o processo de implementação do jogo há eventualidades nas quais a disponibilidade do Game Designer costuma diminuir, como já apontada no *guideline #1*.

O tempo de espera pela disponibilidade do Game Designer nessas ocasiões costuma sofrer variações, bem como a prioridade da tarefa a ser solucionada.

Muitos afirmam que a procura pelo Game Designer ocorre através das ferramentas normais de comunicação, como softwares de *instant messangers, e-mails* e VoIP. Entretanto, o resultado das solicitações depende de uma serie de fatores que variam do tempo que o Game Designer dispõe para a resposta até a forma pela qual a mesma é divulgada.

### Ideal Realista

Sendo assim, foi proposto que para casos como esse, a melhor solução é uma ferramenta (**comentário 5**) colaborativa que possa coletar as dúvidas dos Programadores e organizá-las por prioridade, tal como o software *Bugzilla* **[Bugzilla, 2009]** destinado ao controle de desenvolvimento de sistemas. Dessa forma, o Game Designer pode concentrar seu tempo nas dúvidas cujo caráter emergencial do projeto é mais forte.

Outro beneficio é que por se tratar de uma ferramenta colaborativa, todas as discussões serão de acesso aos envolvidos no projeto, facilitando que outras pessoas consultem o sistema para ver se suas dúvidas já foram solucionadas pela solicitação de outros envolvidos – como em uma FAQ por exemplo.

Além disso, traz outras vantagens como suporte a controle de versões e pode ajudar o Game Designer em futuras versões revisados do DGD.

> ***"...sistema para gerenciar as solicitações/alterações do DGD. Daí O G. Designer trabalha com base nas solicitações dos desenvolvedores" [comentário 5]***

### Ideal Crítico

Apesar do imenso ganho de produtividade que uma ferramenta assim possa criar, nem sempre é possível para as empresas fazer um investimento no desenvolvimento de um sistema destinado a esses propósitos ou até mesmo adquirir um software comercial que traga as funcionalidades corretas de acordo com as necessidades da empresa.

Para casos assim, o mais indicado é usar ferramentas gratuitas da web que permitam edição colaborativa como Blogs **[Blogger, 2009] [Wordpress, 2009]** ou Fóruns, que podem servir de repositórios para dúvidas e diários de produção.

**#5** **Planejamento Prévio Para Resolução de Conflitos**

***Benefícios***

*• focos de orientação para equipe solucionar grande parte dos conflitos de interpretação de forma autônoma.*

***Baixo Custo –*** *a definição prévia do(s) foco(s) de orientação ocorre no inicio do processo de desenvolvimento e perdura até o seu final, permitindo de imediato uma autonomia para a equipe de produção do Jogo.*

É evidente que não há sistema livre de falhas e muito menos um sistema que possa agradar a todos. A produção do software de um jogo assim como a produção de outros tipos de softwares é permeada por conflitos.

No caso da parte técnica, esses conflitos surgem muitas vezes por insuficiência de alguma tecnologia para tratar determinado requisito, seja por falhas da implementação da mesma ou por documentação incompleta.

O grande problema na verdade, segundo os participantes do *survey* diz respeito à experiência que o design do jogo se propõe a oferecer ao jogador. As conseqüências, quando descobertas tardiamente são sempre drásticas implicando em um trabalho extra que pode necessitar do esforço de vários dos profissionais associados ao jogo.

## Ideal Realista

Já que conflitos são inevitáveis, o melhor a se fazer é preparar-se adequadamente para enfrentá-los da forma mais produtiva o possível.

Uma alternativa é tomar medidas para uma reunião com os envolvidos tendo a pauta de tratamento de conflitos como tema. Tal medida é adotada nos processos locais de desenvolvimento de Jogos adaptados do *Scrum* , que usam reuniões semanais (iterativas) para solucionar, entre outras dúvidas, os conflitos da equipe (**comentário 6**).

***"Fazemos reuniões a cada semana. O GD apresenta e a equipe toda discute, fazemos o planejamento e definimos as prioridades" [comentário 6]***

## Ideal Crítico

O grande problema é que a disponibilidade de uma grande quantidade de pessoas em um mesmo horário num mesmo local tem uma probabilidade pequena de ocorrer, principalmente em equipes com grande número de envolvidos.

Então é preciso que haja uma definição prévia sobre a decisão que prevalecerá considerando a natureza do conflito e a urgência de sua resolução.

Vimos que duas abordagens para decisão de conflitos são atualmente utilizadas (**comentários 7 e 8**):

- ***"Normalmente segue-se o GD. Mas quem decidi mesmo é o produtor...porque ele é quem controla o planejamento e as prioridades do time." [comentário 7]***

Foco na produção – essa abordagem toma todas as decisões de conflito baseadas em questões referentes ao cumprimento de prazos e verificações dos *stakeholders*.

- ***"O documento é constantemente adaptado por conta de razões como custo e escopo, mas o poder de decisão é sempre do GD. Ele que é o encarregado de decidir o que é o melhor para o jogo." [comentário 8]***

Foco no Design – essa abordagem toma as decisões baseadas nas experiências as quais o jogo se propõe a transmitir ao jogador.

A escolha da abordagem deve ser uma decisão prévia que se caracteriza como uma orientação para equipe seguir sempre que o Game Designer, o Produtor e outros envolvidos com a gerência do projeto não possam ser contatados para solucionar algum conflito encontrado no DGD. Para definir qual abordagem seguir é preciso que se tome considerações a respeito do escopo, do tempo de projeto, do conhecimento de publico e das ferramentas utilizadas para o desenvolvimento do jogo **[Schell, 2008]**.

Dessa forma, uma equipe que tem projeto orientado ao foco da produção deverá decidir por soluções de programação cujas medidas de tempo e esforço sejam as mais seguras possíveis, ainda que tal decisão implique em algumas percas sugeridas na experiência de jogo proposta pelo DGD.

Já uma equipe que tem projeto orientado ao foco do Design deverá decidir por soluções de programação que permitam uma experiência de jogo mais próxima possível da sugerida no DGD, ainda que tal decisão possa implicar em futuras negociações de prazos e novas avaliações para mais esforço de implementação.

**#6** **Aprender Com as Próprias Experiências**

***Benefícios***

*• o conhecimento adquirido é assegurado para auto-referencia da equipe de produção.*

***Custo Moderado –*** *há muitas ferramentas gratuitas para se desenvolver logs de produção, entretanto criar uma cultura de uso e constante atualização dos registros do desenvolvimento de projetos não é tarefa das mais simples.*

Aprender com as próprias experiências é uma característica associada à maturidade de processos organizacionais **[Biberoglu, 2002]** e de grande importância para melhoria da qualidade dos produtos e da própria segurança de uma equipe de desenvolvimento de uma determinada empresa.

**Gráfico 7: facilidade de implementação relativa ao DGD estar ou não finalizado**


Embora, a maioria (58%) dos participantes tenha optado por um DGD preliminar como a maneira mais fácil de implementar um Jogo **[Gráfico 7]**, vimos nessa questão o quanto a experiência influi na visão que os envolvidos na produção de Jogos têm sobre o uso do DGD em suas atividades.

80% dos membros mais experientes da pesquisa (um quantitativo de 5 com mais de cinco anos de experiência na indústria) responderam que consideram ser mais fácil para os Programadores implementar o jogo a partir de um DGD já finalizado.

Isso mostra que o intervalo entre as definições do DGD e a implementação do software do jogo sofrem também pelo fator de segurança da equipe. Ou seja, aqueles que estão há pouco tempo na industria precisam de muito mais suporte para compreender como implementar as descrições do DGD do que aqueles que têm mais tempo de trabalho.

Isso é óbvio e ficou evidente em algumas das informações fornecidas pelos participantes da pesquisa. Evidente que sempre poderão existir dúvidas, mas a tendência pelo que a pesquisa revelou é que as mesmas vão se tornando cada vez mais refinadas à medida que programadores e game designers se habituam ao processo utilizado (**comentário 9**).

***"à medida que a equipe de desenvolvimento vai absorvendo melhor a idéia do jogo, o Game Designer vai se distanciando gradualmente." [comentário 9]***

O interessante a ser feito é criar uma maneira de coletar esse conhecimento dos mais experientes de forma a poder distribuí-lo entre os menos experientes.

A distribuição eficiente faz com que o aprendizado se torne cada vez mais rápido, isso ajuda aqueles que estão iniciando ou aqueles que estão adaptados a processos de outra empresa.

## Ideal Realista

Uma maneira eficiente de armazenar esse conhecimento é através de uma *wiki* para cada projeto, contendo um histórico das dificuldades encontradas e as formas utilizadas para solucioná-las.

A idéia principal é que essa *wiki* possa ser referenciada a todos os desenvolvedores como um sistema de consulta que possa servir de orientação para que os mesmos erros não sejam recorrentes em outros projetos, criando dessa forma um processo cada vez mais organizado e eficiente, alem de uma cultura própria de autoaprendizado da equipe e conseqüentemente da própria empresa.

## Ideal Crítico

Tal conhecimento só terá utilidade de fato caso o mesmo seja bem descrito e continuamente revisitado na *wiki.* Criar um material assim é relativamente fácil e seu custo é baixo. Entretanto, caso programadores (e outros envolvidos) não se comprometam a atualizar corretamente a *wiki* a gestão de conhecimento estará sempre prejudicada **(comentário 10)**

***"Ferramenta é bom (sic), mas é difícil pra galera usar..." [comentário 10]***

**#7** **Criar DGD Mínimo Para Iniciar Implementação**

***Benefícios***

*•geração de um subconjunto de características suficientes para permitir o inicio do desenvolvimento do software do Jogo.*

***Custo baixo –*** *a pesquisa interna para definir o subconjunto de características é feita apenas uma vez e serve a diversos projetos, sendo necessário apenas atualizar os dados da mesma em períodos definidos pela equipe ou empresa para estar sempre em sintonia com as atualizações do mercado (**ideal critico**).*

A grande maioria dos participantes da pesquisa aponta como um fato comum o desenvolvimento de jogos a partir de um DGD preliminar [**Gráfico 8**].

**Gráfico 8: opiniões acerca da possibilidade de implementar o jogo a partir de um DGD preliminar**


Perguntamos então, qual deveria ser o conteúdo mínimo de tal DGD preliminar para que a equipe de programação possa iniciar o desenvolvimento do jogo.

Dos 16 itens propostos, os mais indicados para poder orientar o inicio do desenvolvimento foram:

- Visão geral (High Concept) do jogo 97%
- Personagens (controláveis ou não pelo jogador) 55%

- Fluxo do game 82%
- Controles (ações do jogador com o personagem) 85%
- Regras do jogo 88%
- Formas de pontuação 61%
- Estilo (ação, estratégia, puzzle) 73%

Os critérios adotados para tal seleção foram os seguintes:

- A quantidade total de itens para criação do DGD preliminar não poderia superar a metade (8) do número total (16) de itens definidos no nosso padrão de DGD;
- Um item apenas poderia figurar na lista caso obtivesse uma porcentagem de aprovação maior que 50%;
- Em caso de empate percentual superando o número mínimo de 8 itens, a escolha entre os elementos seria definida inicialmente pela atividade dos participantes que escolheram tais itens de acordo com a seguinte ordem de prioridade:
  - 1º Programadores, Engenheiros de Software e Lideres Técnicos
  - 2º Produtores
  - 3º Game Designers
  - 4º Artistas
    - Em caso de persistência do empate, o critério utilizado seria a experiência, com ordem de prioridade decrescente.
    - Por fim, o último critério seria a escolha do pesquisador com preferência pelo item que descrevesse a atividade mais próxima de beneficiar o trabalho de implementação.

Tais itens refletem a opinião dos envolvidos na nossa pesquisa e se adéqua ao conhecimento que os mesmos possuem sobre os processos de desenvolvimento por eles executados.

## Ideal Realista

O ideal é que as equipes de desenvolvimento possam elaborar alguma forma de pesquisa interna para definir quais os seus itens do Documento de Game Design preliminar mais importantes para iniciar a implementação do software do jogo, pois cada membro possui uma opinião muito particular a respeito (**comentário 11**).

***"A grosso modo, eu penso que, o DGD deveria ter todas as informações definições do fluxo de tela, das regras (incluindo as definições matemáticas). Não precisa ter arte conceitual e os layouts têm de permitir fácil retrabalho...." [comentário 11]***

## Ideal Crítico

É relevante que tal pesquisa seja reavaliada em períodos recorrentes, pesquisas desse nível revelam uma visualização da realidade atual do conhecimento dos envolvidos com processos de desenvolvimento de jogos. Como o cenário da indústria é bastante intenso no que tange a mudanças é muito importante que tais itens de Documento de Game Design estejam sempre de acordo com os propósitos de desenvolvimento das equipes e das próprias empresas.

**#8** **Criar Artefatos Auxiliares**

***Benefícios***
*•artefatos auxiliares ajudam a interpretar questões mais técnicas do DGD e aumentam produtividade do software do Jogo.*

***Custo moderado –*** *nem sempre é possível produzir todos os artefatos e na riqueza de detalhes indicados para gerar tal auxilio a equipe de desenvolvimento do software.*

Os resultados da pesquisa revelaram que a criação de artefatos complementares ao DGD ajudam a equipe de implementação a manter uma melhor produtividade durante o processo de desenvolvimento **[Gráfico 9**].

**Gráfico 9: níveis de concordância em relação a geração de artefatos complementares ao DGD**


## Ideal Realista

O ideal, segundo os participantes da pesquisa é que sejam gerados artefatos definidos de acordo com as seguintes considerações (**comentário 12, 13, 14, 15**):

- ***"No início do projeto, enquanto os requisitos/projeto do jogo ainda estão mais propensos a mudanças" [comentário 12]***

*Requisitos / Estórias* - As *features* **[Furtado, 2009]** do DGD podem ser mapeadas em *requisitos* ou em *estórias* tomando como referências os modelos já existentes, para esse tipo de documentação na Engenharia de Software, entretanto com os devidos cuidados para não descaracterizar as pretensões contidas pelo DGD.

- ***"Prototipação e ajustes de mecânica (calibração de parâmetros) após testes beta" [comentário 13]***

*Protótipos* - Os protótipos foram citados como ótimas ferramentas para simular a mecânica pretendida nos jogos, pois são desenvolvidos de forma rápida, econômica, fácil de testar e modificar **[Preece, 2004] [Lundgreen, 2008]**. Além disso, possuem uma variedade enorme de usos: coleta de novos requisitos, avaliações de interfaces, apresentações preliminares para *stakeholders*, etc.

- ***"Usamos máquinas de estado para representar ações e eventos dos jogos." [comentário 14]***

*Maquinas de estados* **[Weller, 2009]** - São abstrações utéis para equipe compreender melhor como implementar uma seqüência de telas de jogo ou movimentos de personagens (jogáveis ou não) e o resultado das interações do jogador de forma geral.

- ***"Uma arquitetura UML ajuda, principalmente pra quem possui algum BG (sic) técnico." [comentário 15]***

*Arquitetura* - O documento da arquitetura é uma forma abstrata de descrever técnicamente a produção do jogo muito útil para a equipe de programação, pois ajuda na compreensão dos limites de cada módulo do projeto e como se dará de maneira geral a interação com o usuário **[Furini, 2008]**.
Além disso, pode ser desenvolvida paralelamente de acordo com as mudanças do DGD e facilitar futuramente a produção de jogos semelhantes através de técnicas de reusabilidade.

## Ideal Crítico

Infelizmente, não é em todos os casos que a produção de um jogo poderá contar com toda essa riqueza de detalhes durante as etapas de seu processo, fatores como tempo, escopo, mudanças na equipe podem impedir que artefatos como os citados sejam gerados prejudicando a produtividade do trabalho.

Para isso, acreditamos que pelo menos dois dos quatro itens acima sejam definidos como essenciais pela equipe.

Nossa sugestão é que sejam os itens de *requisitos/estórias* e a *arquitetura*, pois são de familiaridade maior da equipe técnica. Dessa forma, os *protótipos* para avaliação seriam as próprias versões dos jogos e a abstração das *máquinas de estado* seriam substituídas por mais informações nos conjuntos de *requisitos.*

Entretanto, o melhor mesmo é a definição da própria equipe, pois há ganhos e percas independente da escolha e o mais adequado e balanceá-las de acordo com as pretensões do projeto e a experiência do time **[Viller, 2005]**.

**#9** ***Definição Prévia de Avaliações de Jogabilidade***

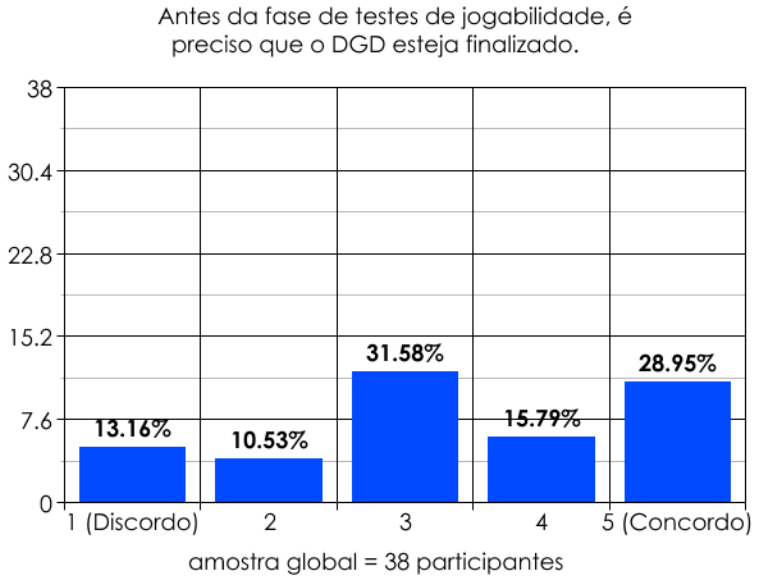
**Benefícios**
•orientação para criação de um artefato paralelo para avaliação da jogabilidade com foco em pesquisas de usuário.

**Custo moderado** -– a infra-estrutura para coletar os resultados de avaliações desse tipo nem sempre são possíveis de se obter. A alternativa recai sobre a metodologia da avaliação, orientada aos recursos disponíveis pela equipe para tal tarefa.

Uma das maiores indefinições dos participantes do *survey* se deu quanto a necessidade de o DGD estar pronto ou não para que se tenha inicio a fase de testes de jogabilidade [**Gráfico 10**].

A maioria dos colaboradores (32%) não conseguiu chegar a uma decisão sobre sua concordância ou não com a afirmação.

**Gráfico 10: níveis de concordância em relação à necessidade de ter o DGD concluído para iniciar testes de jogabilidade.**




De acordo com a análise qualitativa, percebemos que muitos dos participantes se referiam a fase de testes de jogabilidade como uma fase paralela, a qual pode ocorrer em qualquer momento do processo de desenvolvimento desde que haja um designer (preferencialmente o Game Designer) (**comentário 16**) para analisar os resultados e prover os ajustes necessários.

***"...polimentos na jogabilidade são tarefas importantes do design do jogo, já que o GD (Game Designer) tem uma idéia mais precisa de como o jogo deve ser." [comentário 16]***

Entretanto, ocorreram menções referentes ao balanceamento do game relacionada a outras questões que a colocavam mais próxima ao final do processo, quando a grande maioria dos elementos de design, arte e programação já estão mais consolidados e há a existência de uma versão executável muita próxima do que será o jogo quando concluído (**comentário 17**).

***"o GD precisa participar dos ajustes de mecânica (calibração de parâmetros) após testes beta..." [comentário 17]***

Considerando tal disparidade de opiniões e a indefinição por parte da maioria dos participantes, acreditamos que o ideal para esse caso é o uso de uma avaliação de jogabilidade iterativa seguindo os avanços no desenvolvimento do DGD e do proprio software do Jogo.

### Ideal Realista

A melhor forma de conduzir uma avaliação segundo tal modelo de produção é inicialmente a criação de um artefato exclusivo de avaliação que possa incluir o que será avaliado (habilidade de aprendizado, atitudes do jogador, satisfação, etc.) e como serão coletadas tais informações (observação direta, discurso do usuário/jogador, grupos de foco, etc.).

Em seguida, o artefato deverá passar por um teste piloto **[Wacker, 2003]** supervisionado por outra pessoa (da equipe ou externa) que possua conhecimentos na área de Usabilidade para validar a metodologia de avaliação.

Por último, providenciar formas de registrar e distribuir as avaliações remotamente para permitir que os testes possam ser realizados fora do seu local de desenvolvimento (desde que isso não entre em conflito com regras de sigilo sobre produtos da empresa) **[Sedig, 2001]** e acompanhadas pelo Game Designer mesmo que este não possa estar presente no local e no momento do teste.

### Ideal Crítico

Entretanto caso não haja condições para que a equipe realize tal procedimento de infra-estrutura para a avaliação de jogabilidade, recomendamos que pelo menos o cuidado na geração de um artefato exclusivo seja mantido, pois a criação cuidadosa de uma metodologia (quantitativa ou qualitativa) é o fator mais importante de qualquer avaliação **[Batista, 2007][Flick, 2004]** e inclui os fatores necessários para coletar os dados de interesse para a pesquisa.

**#10** **Apresentação do DGD**

***Benefícios***

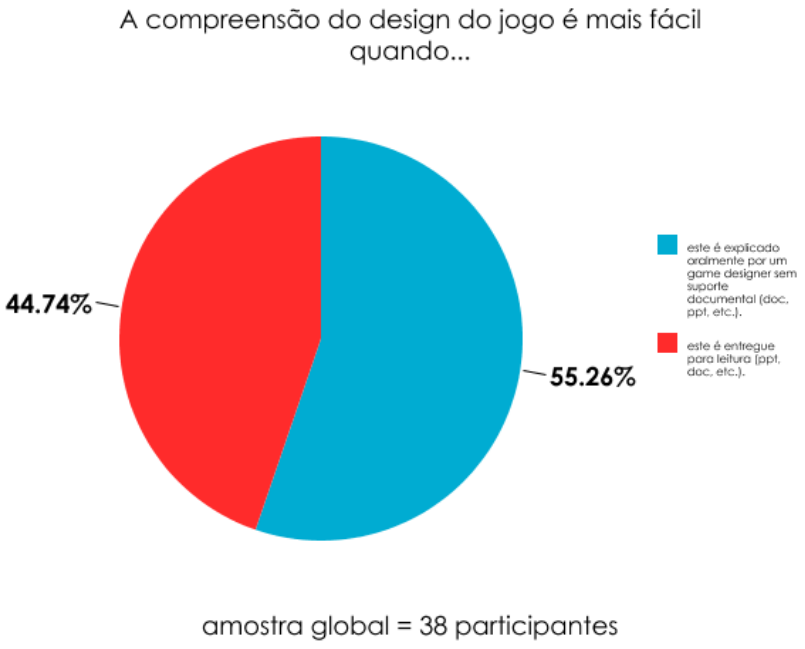
*•mantem o padrão do DGD durante o processo de produção de acordo com os elementos mínimos de um DGD previamente definidos.*
*•facilita controle de versões.*
*•facilita a equipe de programação a se habituar com a linguagem do Game Designer.*

***Custo Baixo -*** *manter o mesmo padrão do documento a partir dos elementos prévios definidos como requisitos mínimos de um DGD é uma atividade bem mais cômoda para o Game Designer que o escalonamento de horários para ter de se reunir com a equipe de programação a cada versão do documento.*

A apresentação de um DGD, ou a apresentação das versões do DGD é um item no processo de jogos estritamente relacionado à comunicação. Por isso mesmo, difícil de se chegar a um consenso devido à quantidade de interpretações que surgem pela heterogeneidade da equipe e dos elementos descritos no DGD.

Observando as respostas de todos os envolvidos na pesquisa, observa-se que apesar de haver certo equilíbrio, há preferência pela explicação oral do Game Designer do que pela leitura do DGD [**Gráfico 11**]. Os participantes do *survey* apontaram essa opção como mais adequada a compreensão do DGD.

**Gráfico 11: opiniões sobre as formas de melhor compreensão do design do jogo**



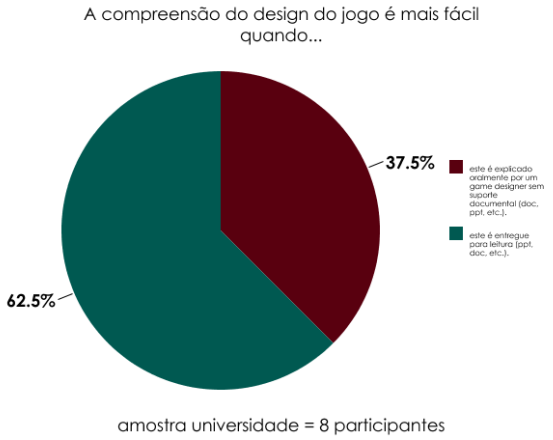

Isso se deve ao intenso processo aos quais os participantes, estão envolvidos. Devido a grande quantidade de atividades em andamento a explicação oral de um Game

Designer sobre o projeto do jogo é mais ágil do que a leitura de um documento, que em alguns casos nem chega a ocorrer (**comentário 18**). Pois permite que dúvidas sejam solucionadas no momento em que são apresentadas facilitando a dinâmica da produção.

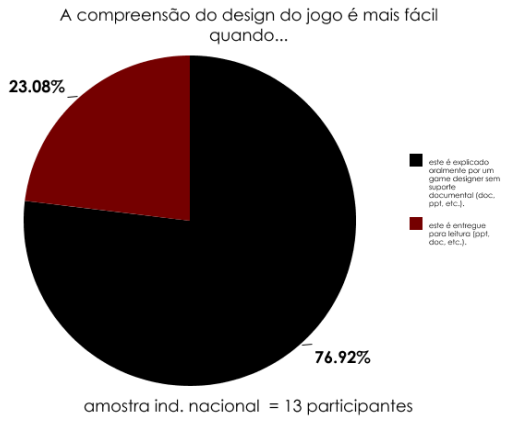
***"normalmente a equipe não lê o GD. muitos não o lêem..." [comentário 18]***

Um fato interessante é que na analise do subgrupo dos participantes que estão na universidade **[Gráfico 12]** ou na indústria internacional **[Gráfico 13],** o resultado foi inverso ao obtido na analise global e no subgrupo dos participantes da indústria nacional **[Gráfico 15].** Nesse caso há uma preferência pela leitura do DGD já a partir do primeiro momento, ou seja, sem a necessidade de uma explicação prévia do Game Designer.

**Gráfico 12, 13 e 14: opiniões sobre as formas de melhor compreensão do design do jogo (à esquerda (12): universidade, à direita (13): indústria internacional, abaixo (14): indústria nacional)**



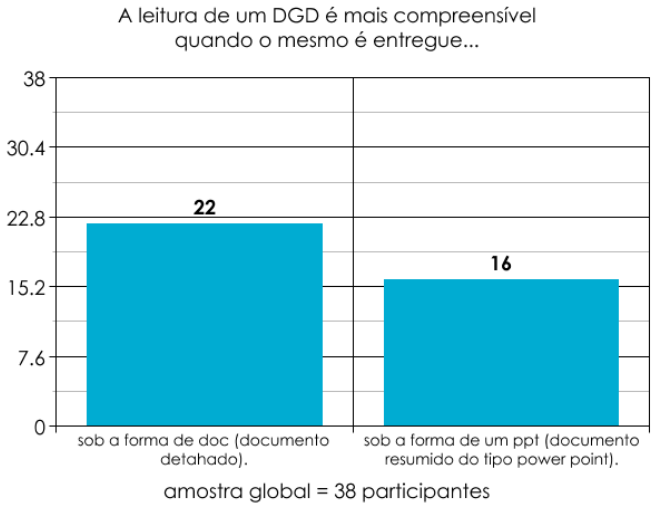





Um dos fatores que contribui com essa preferência do subgrupo formado pelos participantes da universidade é o fato da formação acadêmica condicionar os pesquisadores ao uso em primeira instância da linguagem escrita **[Santa-Clara, 2004]** para a comunicação.

Para o caso da leitura ser indispensável, os participantes consideraram que a melhor forma de compreensão para o DGD é a partir de um documento detalhado, que mantenha uma estrutura mais formal **[Gráfico 16]**.

**Gráfico 16: comparação quanto a melhor forma de se compreender um DGD**




## Ideal Realista

Partindo destes dados considera-se que o ideal é a geração de versões do DGD acompanhadas de um encontro entre Game Designer e equipe de programação para explicação oral do mesmo e solução de problemas de interpretação por parte da equipe (**comentário 19**).

***"fazemos reuniões com GD e equipe para discutir o DGD em termos de prazo, escopo e custos. Isso é bom porque muitos não lêem e dessa forma é mais dinâmico" [comentário 19]***

Cada versão do DGD deve conter as informações necessárias à respeito dos elementos tidos como mínimos para criação de um DGD preliminar (*guideline* 7) para a equipe de implementação.

## Ideal Crítico

O processo de iteração do DGD é constante e nem sempre é possível para o Game Designer e os membros da equipe de programação participarem todos de um mesmo encontro (*guidelines* 5 e 7). Por isso é importante que sejam mantidas sempre a cada versão do DGD as informações dos elementos previamente definidos como essenciais para a implementação a partir da versão preliminar do DGD.

Isso mantém o padrão durante o processo de produção, facilita o controle de versões e permite aos Programadores se habituarem à linguagem do Game Designer diminuindo a dependência de explicações extras por parte do mesmo.

## #00 DGD Finalizado – Casos Especiais

Consideramos na nossa pesquisa o caso especial de poder desenvolver o Jogo apenas com o DGD finalizado. Perguntamos se tal procedimento seria adequado e as respostas se dividiram completamente. Na amostra global, dos 38 participantes 50% concordaram e 50 % discordaram [**Gráfico 16**].

**Gráfico 16: respostas, considerando a visão de produtor, acerca de ser válido finalizar um DGD antes da implementação**


O que foi afirmado é a existência de uma comodidade para o desenvolvimento do processo caso o DGD esteja finalizado (**comentário 20**).

"***Em todos os casos é benéfico para todo o processo que o DGD esteja finalizado antes do início do desenvolvimento. Mesmo que seja possível fazê-lo caso isto não ocorra, em todos os casos vale à pena ter o mesmo finalizado. É mais uma questão de viabilidade.***" ***[comentário 20]***

Entretanto, um comentário (**comentário 21**) muito semelhante foi realizado, mas com ressalvas a aplicação prática de se implementar um Jogo apenas se o DGD estiver concluído devido à natureza mutante do processo.

***"Sempre vai valer a pena ter um DGD pronto e finalizado antes de começar a implementação, mas honestamente isso é bem utópico. O DGD é um documento fluido, que se altera conforme o jogo progride e novos limites são impostos, o que faz com que ele quase nunca permaneça o mesmo que era no início do desenvolvimento do jogo." [comentário 21]***

Dado o nível da divergência, completamos a discussão perguntado em quais casos é imprescindível que o DGD esteja finalizado. Os resultados estão na **[Tabela 3]**

**Tabela 3: dados da amostra global sobre os casos imprescindíveis para um DGD estar finalizado.**

| | |
|---|---|
| Outsourcing (o jogo vai ser desenvolvido por uma equipe terceirizada e remota) | 76% |
| Equipe remota (o jogo vai ser desenvolvido por uma equipe da empresa porém remota) | 57% |
| Jogo complexo do tipo AAA (Halo, Shadow of the Colossus, etc.) | 71% |
| Equipe de desenvolvimento grande | 67% |
| Outros | 24% |

**Obs.: os participantes podiam selecionar mais de uma opção desta lista**

De forma geral, os participantes argumentam que quanto mais terceirizados ou remotos forem as requisições será melhor para o processo ter um DGD finalizado, bem como em casos de Jogos AAA nos quais as equipes de desenvolvimento costumam ter grande numero de participantes que desenvolvem títulos de acordo com padrões já estabelecidos da industria**[Højsted, 2006]**

Outros casos citados foram com relação a projetos de Jogos específicos para testes de alguma nova tecnologia ou gêneros não convencionais.

# 6. Conclusões

## 6.1 Visão Geral dos Processos

De uma maneira geral, considerando os dados obtidos com a amostra geral da pesquisa percebesse que o processo do desenvolvimento de Jogos que permita uma maior conexão entre o Design e o software deve possuir características mais próximas das metodologias Ágeis, contando com a presença do Game Designer em todo o processo e com a produção iterativa do DGD, contendo suporte de documentos auxiliares que expliquem com detalhes mais técnicos informações necessárias à equipe de programação.

## 6.2 Planos Contingenciais

Importante também e a criação de planos contingenciais que permitam à equipe de desenvolvimento possuir certa autonomia para solucionar conflitos em momentos nos quais a presença do Game Designer no processo não seja garantida.

## 6.3 Documentação

Apesar dos processos adotados pela indústria considerarem uma abordagem Ágil para o desenvolvimento de Jogos, percebemos que a característica de pouca documentação das metodologias Ágeis não implica em mais produtividade como proposto inicialmente pelo manifesto Ágil (http://agilemanifesto.org/). No caso dos dados obtidos de acordo com a amostra deste trabalho ocorreu exatamente o contrário.

Como há um intenso trabalho de interpretação das informações contidas no DGD, a produção de documentos que possuam detalhe mais técnicos como a especificação dos Requisitos ou da Arquitetura auxiliam a produção de um melhor Jogo, pois reduzem os esforços de Programadores em compreender certas definições do Design do Jogo e diminuem a ocorrência de dúvidas. De forma geral, vimos que a criação de documentos auxiliares é uma boa prática não apenas para a produção do software do Jogo, mas também para outras etapas da produção como as fases de testes, testes de usabilidade e avaliação de Jogabilidade.

# 7. Trabalhos Futuros

## 7.1 Revisão dos *Guidelines*

Entendemos como primeiro esforço na continuação deste trabalho a revisão dos *guidelines* a partir da avaliação dos participantes envolvidos na pesquisa. Tal revisão tem o objetivo de identificar quais os *guidelines* de maior utilidade e melhorar aqueles que por alguma eventualidade não tenham conseguido alcançar as expectativas dos colaboradores.

## 7.2 Adaptação Continua dos *Guidelines* aos Processos

Os processos de desenvolvimento de Jogos, naturalmente estão sempre buscando maior ganho de produtividade. Dessa forma, é importante ter conhecimento de como os *guidelines* podem ser inseridos em tais processos facilitando tal objetivo e sendo constantemente atualizados de acordo com as necessidades das equipes de desenvolvimento.

## 7.3 Nova Análise dos Dados

A quantidade de dados gerados pela pesquisa foi bastante satisfatória e uma nova análise pode ser realizada a partir dos mesmos para criação de novas categorias para reforçar as informações concluídas ou encontrar novas considerações tanto em caráter quantitativo quanto qualitativo.

## 7.4 Participações do Usuário

Não observamos na pesquisa realizada menção a participação dos usuários (jogadores) nos processos de desenvolvimento. Apenas uma das empresas se pronunciou sobre tal dúvida revelando que equipes remotas cuidam dos detalhes concernentes a avaliações de interfaces, Design Centrado no Usuário e *Reply Value*. Dado este fato, entendemos que *guidelines* que envolvam a presença do usuário são estritamente necessários.

## 7.5 Arte e Implementação

A arte do Jogo descrita no Documento de Game Design também necessita de um planejamento a parte para poder integrar o software do Jogo de forma a permitir que a execução aconteça sem problemas derivados do conteúdo artístico. Para isso, um trabalho semelhante a este, entretanto com foco ao desenvolvimento de software orientado ao material de produção artística.

# 8. Referências


**[Calele, 2005]**

David Callele , Eric Neufeld , Kevin Schneider, *Requirements Engineering and the Creative Process in the Video Game Industry*, Proceedings of the 13th IEEE International Conference on Requirements Engineering, p.240-252, August 29-September 02, 2005 [doi>10.1109/RE.2005.58]

**[Rodriguez, 2007]**

Ricardo Ruiz Rodríguez, Carlos Alberto Fernández-y-Fernández, *Towards a Design Process for Didactic Game Development: experiences and Proposals of the Edumóvil project. (2007),* Morelia, Michoacan Publisher: IEEE

**[Game Developer Magazine, 2009]**

Game Developer Magazine, 2009, www.gdmag.com

**[Bergan, 2007]**

Bergan, Ronald. *Guia Ilustrado Zahar cinema* / Ronald Bergan; Tradução: Carolina Alfaro. – Rio de Janeiro: Jorge Zahar Ed. , 2007.

**[Field, 2001]**

Field, Syd. *Manual do roteiro: os fundamentos do texto cinematográfico* / Syd Field. - Rio de Janeiro: Objetiva, 2001 ISBN 85-7302-044-X

**[Calele, 2006]**

David Callele , Eric Neufeld , Kevin Schneider, *Emotional Requirements in Video Games,* Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06), p.292-295, September 11-15, 2006 [doi>10.1109/RE.2006.19]

**[Carvalho, 2006]**

Carvalho G. H. P. *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. 2006.

**[Alves, 2007]**

Alves, C. Ramalho, G. Damasceno, A. *Challenges in Requirements Engineering for Mobile Games Development: The Meantime Case Study*. Univ. Fed. de Pernambuco, Recife;

**[Furtado, 2009]**

Furtado, André, W. B. *Empowering Digital Games Development through Software Factories and Domain-specific Language: an End-to-End Approach*. Tese de Doutorado. Pós-Graduação em Ciência da Computação. Centro de Informática. Universidade Federal de Pernambuco. 2009.

**[Araujo, 2009]**
ARAUJO, Allan. *"Play4Fun: Uma Fábrica de Jogos Digitais Casuais"*. Dissertação de Mestrado. Pós-Graduação em Ciência da Computação. Centro de Informática. Universidade Federal de Pernambuco.2009.

**[Schuytema, 2006]**
Scuytema, Paul. *Game Design: A Practical Aproach*. Charles River Media; 1 edition (July 27, 2006)

[**Schell, 2008**]
Schell, J., 2008. *Art of Game Design: A Book of Lenses.Burlington*: Elsevier

**[Rollings, 2003]**
Rollings, Andrew. Adams, Ernest. *Andrew Rollings and Ernest Adams on Game Design (Paperback).* New Riders Games; Ltd Rmst edition (May 11, 2003)

**[Porto Digital, 2009]**
Porto Digital, 2009, www.portodigital.org/

**[UFPE, 2009]**
Universidade Federal de Pernambuco – UFPE, 2009, www.ufpe.br

**[Ceruzzi, 2003]**
Ceruzzi, Paul E. *A history of modern computing* / Paul E. Ceruzzi.—2nd ed. 1998, 2003 Massachusetts Institute of Technology.

**[Sutherland, 2003]**
Ivan E. Sutherland, *Sketch pad a man-machine graphical communication system*, Technical Report, University of Cambridge, September 2003

**[Sommerville, 2000]**
Sommerville, Ian. *Software Enginner (6th Edition).* Addison Wesley; 6 edition (August 21, 2000)

**[Rational, 2009]**
IBM Rational Software, 2009, http://www01.ibm.com/software/rational/. Acessado em: Novembro 2009.

**[Carrol, 2005]**
Edward R. Carroll, *Estimating software based on use case point*s, Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, October 16-20, 2005, San Diego, CA, USA [doi>10.1145/1094855.1094960]

**[GameDev, 2009]**
GameDev.Net, 2009, http://www.gamedev.net/reference/articles/article1940.asp. Acessado em: Novembro 2009.

**[Agile Game Development, 2009]**
Agile Game Development, 2009, http://www.agilegamedevelopment.com/. Acessado em: Novembro 2009.

**[IGDA]**
International Game Developers Association – Recife, 2009, http://www.igda.org/recife. Acessado em: Novembro 2009.

**[Sedig, 2001]**
Kamran Sedig , Maria Klawe , Marvin Westrom, Role of interface manipulation style and scaffolding on cognition and concept learning in learnware, ACM Transactions on Computer-Human Interaction (TOCHI), v.8 n.1, p.34-59, March 2001

**[Young Oh, 2006]**
Ji-Young Oh, Wolfgang Stuerzlinger, John Danahy, SESAME: Towards Better 3D Conceptual Design Systems, Proceedings of the 6th conference on Designing Interactive systems, 2006, University Park, PA, USA

**[Shin, 2007]**
HyoJong Shin , Takeo Igarashi, Magic canvas: interactive design of a 3-D scene prototype from freehand sketches, Proceedings of Graphics Interface 2007, May 28-30, 2007, Montreal, Canada [doi>10.1145/1268517.1268530]

[**NVIVO, 2009]**
NVIVO, 2009, http://www.qsrinternational.com/products_nvivo.aspx. Acessado em: Novembro 2009.

**[Flick, 2004]**
FLICK, Uwe. Uma introdução à pesquisa qualitativa**.** 2.ed. Porto Alegre: Bookman, 2004. 312 p

**[Google Documents, 2009]**
Google Documents, 2009, docs.google.com

**[NCES, 2009]**
National Center for Educational Statistics - NCES, 2009, http://nces.ed.gov/nceskids/. Acessado em: Novembro 2009.

**[Sommerville, 1997]**
Sommerville, Ian. Sawyer, Peter. Requirements Engineering: A Good Practice Guide (Paperback). Wiley (April 28, 1997).

**[Wetzel, 2008]**
Wetzel, R., McCall, R., Braun, A., and Broll, W. 2008. Guidelines for designing augmented reality games. In *Proceedings of the 2008 Conference on Future Play:*

*Research, Play, Share* (Toronto, Ontario, Canada, November 03 - 05, 2008). Future Play '08. ACM, New York, NY, 173-180.

**[The Eye of Judgement, 2009]**
The Eye of The Judgement, 2009, http://www.us.playstation.com/EyeofJudgment/. Acessado em: Novembro 2009.

**[PlayStation 3, 2009]**
Sony Playstation 3, 2009, http://www.us.playstation.com/. Acessado em: Novembro 2009.

**[Grammenos, 2008]**
Grammenos, D. 2008. Game over: learning by dying. In *Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems* (Florence, Italy, April 05 - 10, 2008). CHI '08. ACM, New York, NY, 1443-1452. DOI= http://doi.acm.org/10.1145/1357054.1357281

**[Ramachadran, 2005]**
Ramachandran, M. 2005. Software reuse guidelines. *SIGSOFT Softw. Eng. Notes* 30, 3 (May. 2005), 1-8. DOI= http://doi.acm.org/10.1145/1061874.1061889

**[Baxter, 1998]**
Baxter, M. 1998. Projeto de Produto: Guia Prático Para Design De Novos Produtos. Tradução De Chapman & Hall, 5.Ed. Blucher.

**[Mycoted, 2009]**
Mycoted, 2009, www.mycoted.com. Acessado em: Novembro 2009.

**[Disney, 2009]**
Walt Disney Studios, 2009, disney.com. Acessado em: Novembro 2009.

**[Moløkken-Østvold, 2008]**
Moløkken-Østvold, K., Haugen, N. C., and Benestad, H. C. 2008. Using planning poker for combining expert estimates in software projects. *J. Syst. Softw.* 81, 12 (Dec. 2008), 2106-2117. DOI= http://dx.doi.org/10.1016/j.jss.2008.03.058

**[Bugzilla, 2009]**
Bugzilla, 2009, http://www.bugzilla.org/. Acessado em: Novembro 2009.

**[Blogger, 2009]**
Blogger, 2009, www.blogger.com/. Acessado em: Novembro 2009.

**[Wordpress, 2009]**
Wordpress, 2009, wordpress.org/. Acessado em: Novembro 2009.

**[Preece, 2004]**

Kurniawan, S. 2004. Interaction design: Beyond human–computer interaction by Preece, Sharp and Rogers (2001), ISBN 0471492787. *Univers. Access Inf. Soc.* 3, 3 (Oct. 2004), 289-289. DOI= http://dx.doi.org/10.1007/s10209-004-0102-1

**[Lundgreen, 2008]**

Sus Lundgren, Designing games: why and how, ACM: New York, USA, 2008

**[Lewis, 2009]**

Lewis, J. R. and Sauro, J. 2009. The Factor Structure of the System Usability Scale. In *Proceedings of the 1st international Conference on Human Centered Design: Held As Part of HCI international 2009* (San Diego, CA, July 19 - 24, 2009). M. Kurosu, Ed. Lecture Notes In Computer Science, vol. 5619. Springer-Verlag, Berlin, Heidelberg, 94-103. DOI= http://dx.doi.org/10.1007/978-3-642-02806-9_12

**[Furini, 2008]**

Furini, M. 2008. An architecture to easily produce adventure and movie games for the mobile scenario. *Comput. Entertain.* 6, 2 (Jul. 2008), 1-16. DOI= http://doi.acm.org/10.1145/1371216.1371222

**[Wacker, 2003]**

Markus Wacker , Stanislav L. Stoev , Michael Keckeisen, Wolfgang Straßer, A comparative study on user performance in the Virtual Dressmaker application, Proceedings of the ACM symposium on Virtual reality software and technology, October 01-03, 2003, Osaka, Japan

**[Batista, 2007]**

Batista, Makilim Nunes., Corrêa de Campos, Dinael.Metodologias de Pesquisa em Ciências: Análises Quanitativa e Qualitativa. Rio de Janeiro: LTC, 2007

**[Biberoglu, 2002]**

Biberoglu, E. and Haddad, H. 2002. A survey of industrial experiences with CMM and the teaching of CMM practices. *J. Comput. Small Coll.* 18, 2 (Dec. 2002), 143-152.

**[Weller, 2009]**

Weller, M. P., Do, E. Y., and Gross, M. D. 2009. *State machines are child's play: observing children ages 9 to 11 playing Escape Machine.* In *Proceedings of the 8th international Conference on interaction Design and Children* (Como, Italy, June 03 - 05, 2009). IDC '09. ACM, New York, NY, 170-173. DOI= http://doi.acm.org/10.1145/1551788.1551819

**[Viller, 2005]**

Viller, S., Brereton, M., Redhead, F., and Axup, J. 2005. A collaborative digestion and design game for community and technology exploration. In *Proceedings of the 2005 Conference on Designing For User Experience* (San Francisco, California,

November 03 - 05, 2005). Designing For User Experiences, vol. 135. AIGA: American Institute of Graphic Arts, New York, NY, 47.

**[Santa – Clara, 2004]**

Angela Maria Santa – Clara/ Ticia Cassiany Ferro/ Sandra Patrícia Ataíde Ferreira. *O Papel da Linguagem do Pesquisador na Construção da Compreensão de um Texto.* Estudos de Psicologia. Maio-ago. UFRN: Natal, Brasil

**[Højsted, 2006]**

Anders Højsted and Stein Llanos**,** *"Big Game: Formalizing Test Methods for Computer Game Concepts"* , IT-University of Copenhagen, 161 pages.