MARCOS SILVANO ORITA ALMEIDA

MGUP: RUP APLICADO A JOGOS MÓVEIS

MARINGÁ

2006

Livros Grátis

http://www.livrosgratis.com.br

Milhares de livros grátis para download.

MARCOS SILVANO ORITA ALMEIDA

MGUP: RUP APLICADO A JOGOS MÓVEIS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador:

Prof. Dr. Antonio Mendes da Silva Filho

Co-Orientadora:

Prof. Drª. Elisa Hatsue Moriya Huzita

MARINGÁ

2006

MARCOS SILVANO ORITA ALMEIDA

MGUP: RUP APLICADO A JOGOS MÓVEIS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Aprovado em 24/03/2006.

BANCA EXAMINADORA

Prof. Dr. Antonio Mendes da Silva Filho
Universidade Católica de Pernambuco – DEI/UNICAP

Profª. Drª. Elisa Hatsue Moriya Huzita
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Sérgio Roberto Pereira da Silva
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Fábio Nogueira de Lucena
Universidade Federal de Goiás – INF/UFG

RESUMO

Nas últimas décadas pôde-se observar um crescimento gigantesco em torno do mercado e da tecnologia dos jogos de computador. Os processos e tecnologias envolvidas no desenvolvimento de jogos têm estado em constante evolução desde sua concepção. O Processo Unificado da Rational pode ser considerado principal ponto de partida para vários processos de jogos usados com sucesso na indústria. No entanto, características de determinados domínios podem tornar necessária uma análise mais profunda, partindo dos princípios do processo e de uma discussão sobre todos os seus elementos. Este trabalho tem como objetivo principal pesquisar, propor e documentar um estudo de aplicação do RUP para o desenvolvimento de jogos móveis (de celular). O MGUP – Processo Unificado para Jogos Móveis (*Mobile Game Unified Process*) – é o principal resultado desta investigação. Ele propõe adequações que partem das características fundamentais do processo. Desta forma, concebeu-se uma metodologia orientada à Criação de Jogos (*Game Design*), centrada em arquitetura de software e no *core gameplay* (jogabilidade fundamental) e, essencialmente iterativa e incremental. No MGUP, todos os papéis, atividades, artefatos, fases e disciplinas envolvidas no processo de produção de jogos móveis são discutidos sob a ótica do RUP. Como resultados, uma disciplina de Criação de Jogos é inserida no processo, assim como os papéis, atividades e artefatos referentes ao desenvolvimento de jogos. Alguns modelos e exemplos de artefatos estão presentes como anexos.

Palavras-chave: Engenharia de Software, Processo de Software, Rational Unified Process, Desenvolvimento de Jogos, Jogos para Celular.

ABSTRACT

The market and technology of computer games had experienced a significant growth during the last decades. The processes and technologies used for developing games have evolved almost constantly since the very first days. Together with such efforts, the Rational Unified Process may be considered as a starting point for various game processes successfully used within the industry. However, characteristics of a set of domains may require a deeper analysis, starting from the process principles as well as from all over process elements. This dissertation aims at investigating, documenting and making a proposal of how the Rational Unified Process can be applied to mobile games development. The MGUP - Mobile Game Unified Process is envisaged as the main result of this investigation. The resulting MGUP brings suggestions of how pieces of the original process can be adapted to meet mobile games requirements. In this context, a methodology guided through the game design, centered on software architecture along with core game play, and iterative and incremental has been proposed. For MGUP, all the rules, activities, artifacts, phases, and disciplines involved in the mobile games production process are analyzed under the RUP perspective. As a result, the discipline of Game Design as well as the roles, activities and artifacts needed for games development have been added to the process. A set of templates and artifact examples are used to illustrate the MGUP presentation.

Keywords: Software *Engine*ering, Software Process, Rational Unified Process, Games Development, Mobile Games.

LISTA DE ILUSTRAÇÕES

GLOSSÁRIO

ABNT	Associação Brasileira de Normas Técnicas.
ARM	*Advanced Risc Machine*. Arquitetura de Processadores usados em dispositivos móveis.
Builds	Montagens ou lançamentos do software.
CDMA	Tecnologia de comunicação de redes de telefones celulares.
Console	Dispositivo usado para jogar Videogames. Exemplos: Playstation 2, Xbox e GameCube.
Engine	Motor do Jogo: parte fundamental de seu software.
FDD - *Feature-Driven Development*	Abordagem ágil de desenvolvimento baseado em recursos do software.
Framework	Arcabouço.
GC	Game Cube: Console de videogames da Nintendo. O menos popular do mercado. Concorrente do PS2 e Xbox.
Guidelines	Diretrizes de processo.
I.A.	Inteligência Artificial aplicada a jogos.
Core gameplay	Parte fundamental do *gameplay*, que determina o estilo e os diferenciais das mecânicas de interações de um jogo.
Gameplay	"Jogabilidade". Conjunto de mecânicas que definem a forma de interação do usuário com todos os elementos do jogo.
Jogos Móveis	Jogos para dispositivos móveis. Neste contexto, refere-se a jogos para aparelhos de telefone celular.
MGUP	Processo Unificado para Jogos Móveis (*Mobile Game Unified Process*). Proposta de processo baseado no RUP para desenvolvimento de Jogos de Celular.
NPC	*Non Player Character*. Personagens controlados pelo jogo.
Objectory Process	Processo de desenvolvimento de software baseado em objetos. Antecessor ao Processo Unificado da Rational.
PC	Computador Pessoal

PC-Game	Jogos específicos para computador.
Portáteis	Consoles portáteis de videogames. Exemplos: Game Boy Advance, Nintendo DS e Sony PSP.
PS2	Playstation 2: Console de videogames da Sony. O mais popular do mercado de consoles.
Release	Lançamento de versão funcional do software.
RUP	Processo Unificado da Rational (*Rational Unified Process*).
RUP Lite	O mesmo que RUP Light: “RUP Leve”.
SMS	*Short Message Service*. Mensagens de texto enviadas para celulares.
Template	Modelo de documento.
UML	Unified Modeling Language.
Videogames	Refere-se a Jogos Computadorizados.
Xbox	Console de videogames da Microsoft. O principal concorrente do PS2.
XP - *Extreme Programming*	Programação Extrema: Abordagem ágil de desenvolvimento de software.
Protótipo Físico	Implementação dos conceitos de um jogo na forma de jogo de tabuleiro. Ajuda em testes de balanço de variáveis do jogo.
Protótipo em Software	Implementação dos conceitos de um jogo em software. O protótipo contém alguns dos elementos do jogo.

SUMARIO

1 Introdução

1.1 Contextualização

Os jogos eletrônicos passaram de pequenos projetos a grandes e sofisticadas aplicações que levam anos para serem concluídas e demandam o esforço coletivo de dezenas de profissionais das mais diversas áreas. Essa característica multidisciplinar traz desafios e complexidades diferentes das encontradas em outros produtos de softwares.

Atualmente o entretenimento digital faz parte da vida de milhões de pessoas. Computadores, sofisticados consoles, videogames portáteis, telefones celulares, Internet e aparelhos de DVD: os jogos estão cada vez mais presentes no cotidiano. A cada momento, novas tecnologias abrem possibilidades de pesquisa e desenvolvimento.

Na última década, houve um gigantesco crescimento do entretenimento digital [ESA 2005], mas infelizmente a produção e o mercado de jogos no Brasil estão ainda distantes de países como Japão e Estados Unidos.

A computação móvel (especialmente os telefones celulares) está crescendo rapidamente, mas ainda pode ser considerada uma área recente. Empresas como Nokia e Motorola têm feito grandes investimentos para contribuir com a evolução da tecnologia, instalando bases de pesquisa e desenvolvimento no país.

1.2 Motivação

O desenvolvimento de jogos tem sido bastante explorado em vários países. Exemplos de locais onde o processo e as tecnologias envolvidas estão sendo aperfeiçoados há décadas são: Japão, Estados Unidos, Canadá, Inglaterra e Alemanha. No Brasil, a área sempre foi alvo de pequenos desenvolvedores independentes, mas nos últimos anos, um interesse maior de fábricas de software nacionais tem surgido, embora de forma ainda lenta. Fatores como

inexperiência e falta de pessoal bem preparado têm sido determinantes para o retardo deste crescimento.

O Processo Unificado da Rational é o principal ponto de partida para vários processos de jogos usados com sucesso na indústria ([PEDERSEN 2003], [BETHKE 2003], [GAMEDEV 2005] e [GAMASUTRA 2005]). Sabe-se que existem características bastante específicas quando se considera a produção de jogos para celular (jogos móveis). Estas características podem tornar necessária uma análise profunda do processo, partindo dos pontos fundamentais do RUP.

1.3 Objetivos

A partir das constatações apresentadas, este trabalho de dissertação tem como objetivo principal pesquisar, propor e documentar um estudo de aplicação do Processo Unificado da Rational para o desenvolvimento de jogos de celular, o MGUP – Processo Unificado para Jogos Móveis (*Mobile Game Unified Process*) –, partindo dos princípios fundamentais do processo. Em decorrência disto, espera-se possibilitar melhorias na execução e nos resultados de projetos de desenvolvimento de jogos para celular e, ao mesmo tempo, cultivar a pesquisa e produção de jogos no Brasil.

O MGUP apresenta uma adequação do processo de desenvolvimento de jogos à abordagem RUP, considerando peculiaridades de projetos de jogos móveis. Com isso, os princípios fundamentais, eixos, papéis, atividades, artefatos, fases e disciplinas do RUP são revistos e adaptados para melhor se enquadrarem às necessidades do domínio de jogos. Uma análise sobre a adoção de práticas ágeis da disciplina *Extreme Programming* (Programação Extrema - XP) é realizada, a fim de permitir adaptações a diferentes cenários de projetos de jogos móveis.

O MGUP não é simplesmente o resultado de um estudo de configuração do Processo Unificado da Rational. O MGUP representa uma investigação que parte dos princípios do

RUP e examina todos os elementos necessários a um processo de jogos. O primeiro passo em direção ao MGUP é o RUP Lite (RUP Leve), processo baseado nos princípios fundamentais do RUP e no conjunto de suas principais diretrizes (*guidelines*), chamado de "Essência do RUP". O processo de desenvolvimento de jogos é estudado e peculiaridades da produção de jogos móveis são levantadas. O resultado é analisado frente ao RUP Lite, chegando-se então ao MGUP.

Como conseqüência do objetivo principal, esta investigação tem alguns objetivos secundários, envolvendo partes específicas da temática abordada: desenvolvimento de jogos, processo de software e produção de aplicações móveis. Os principais objetivos secundários identificados foram:

- Compreender melhor o Processo Unificado da Rational, procurando elementos e definições fundamentais que tornassem possível uma aplicação mais adequada ao domínio de jogos móveis (para celular).
- Conhecer metodologias, ferramentas e tecnologias empregadas no desenvolvimento de aplicações para dispositivos móveis.
- Investigar o perfil de processo encontrado na execução de projetos de jogos móveis.
- Investigar o processo de produção de jogos e o processo de Criação de Jogos (*Game design*).

1.4 Organização do Texto

A distribuição dos assuntos e capítulos seguiu a mesma ordem de execução das investigações que levaram à produção deste trabalho.

O Capítulo 2 apresenta uma fundamentação do RUP, ponto de partida da investigação. Somados a este, os próximos dois capítulos constituem subsídios necessários à resposta da pergunta: "Qual o cenário de RUP que possa oferecer suporte a jogos?". Desta forma, o Capítulo 3 apresenta uma discussão em torno da ciência e da arte envolvidas no

desenvolvimento de jogos e o Capítulo 4, uma análise sobre as características comumente percebidas em projetos de desenvolvimento de jogos móveis. A segunda parte do Capítulo 4 representa a resposta à pergunta acima: o RUP Lite e sua relação com outros processos são discutidos.

O Capítulo 5 contém o texto mais relevante desta investigação: a apresentação do MGUP. Ele representa a resposta à segunda pergunta realizada ao longo da investigação: “Como o RUP Lite pode ser melhor aplicado a Jogos Móveis?”.

No final do trabalho, o Capítulo 6 apresenta uma experimentação (e seus resultados) do MGUP frente a um estudo de caso de projetos reais desenvolvidos no C.E.S.A.R. – Centro de Estudos e Sistemas Avançados do Recife. Os projetos foram escolhidos por apresentarem variabilidade em características-chave para este estudo: processo, plataforma de aplicação móvel, linguagem de programação e, tecnologia e paradigma visual. Desta forma, pôde-se avaliar o MGUP frente a realidades de produção distintas. A última parte deste capítulo apresenta os resultados da experimentação e da investigação como um todo, além de apontar trabalhos futuros.

2 RUP: Processo Unificado da Rational

2.1 Introdução

O Processo Unificado da Rational (*Rational Unified Process* ou RUP) é o resultado do esforço conjunto de vários cientistas e profissionais da computação. Atualmente, ele ainda pode ser considerado o principal modelo de processo adotado em muitas empresas de desenvolvimento de software. O mesmo pode ser dito sobre a indústria de jogos.

Este capítulo apresenta uma visão geral do Processo Unificado da Rational (RUP), contemplando seus três princípios, sua essência e as estruturas dinâmica e estática do processo. Uma apresentação mais detalhada estaria fora do escopo deste trabalho.

2.2 Processo de Software

Mesmo hoje, existe a crença de que grandes empresas sejam totalmente organizadas e funcionam simplesmente com base na competência de seus colaboradores. Pensa-se que essas pessoas conhecem todo o trabalho a ser feito e o cumprem sem maiores problemas, de uma forma rotineira e quase automática. Essa crença é em muitos casos enganosa. Sem dúvida, muitos profissionais de software são altamente capacitados e treinados para desempenharem seu papel, mas a profissão pode ser considerada ainda jovem. Como conseqüência, existe a necessidade de tornar o processo de desenvolvimento mais sistemático, semelhante ao que já acontece em outras engenharias.

A necessidade por um processo que leve a padrões de desenvolvimento de software se torna fundamental em companhias e organizações em que o uso de sistemas é crítico. O setor financeiro, controle de tráfego aéreo e sistemas de defesa ou telecomunicações, são exemplos que demonstram a necessidade de se trabalhar com um padrão que ajude a alcançar um resultado confiável, robusto e adequado. Muitas vezes, o sucesso na conduta dos negócios ou

na execução de missões dessas organizações depende de seus sistemas de software.

2.2.1 Definição de Processo de Software

Um processo define "quem" está fazendo, "o que", "quando" e "como", a fim de alcançar um objetivo ([JACOBSON et. al. 1999]). Na engenharia de software, o objetivo é construir um novo sistema ou melhorar algum já existente. Um processo oferece diretrizes e guias para um trabalho em equipe mais eficiente. A idéia é promover uma visão e cultura comum em torno da forma pela qual o software é desenvolvido.

Faz-se necessário que um processo sirva de guia para todos os seus participantes - clientes, usuários, desenvolvedores e gerentes. Também é importante que seja documentado e que os clientes possam compreender sua progressão. O processo deve não só adaptar-se às necessidades e à realidade das equipes, como também, ser “modularizável” e “incrementável”. Desta forma, deve ser compatível com a realidade que as tecnologias, ferramentas, pessoas e padrões permitam. Todos esses fatores são determinantes para o conceito, uso e evolução do processo ([KRUCHTEN 2000]).

Um processo de software pode ser definido como sendo o conjunto de todas as atividades relacionadas ao desenvolvimento, controle, validação e manutenção de um software operacional. Estas atividades incluem: a determinação e especificação dos requisitos do software, a elaboração de protótipos, o projeto do software, a implementação, a verificação e validação do software, o controle de qualidade, a integração dos componentes, a documentação, a gestão de configurações e versões, a gerência do projeto e a avaliação do produto de software ([BECK 2004]

BECK, Kent; ANDRES, Cynthia. **Extreme Programming Explained:** Embrace Change. 2. ed. Addison-Wesley, 2004.

[GIMENES 1994]).

O processo de software abrange tanto as atividades técnicas quanto as gerenciais. As atividades utilizam métodos, ferramentas de desenvolvimento e recursos (por exemplo:

pessoal, financeiro e maquinário). As atividades do processo podem estar relacionadas aos produtos (intermediários ou finais), às estruturas organizacionais e suas restrições. Essas estruturas organizacionais envolvem um trabalho cooperativo entre pessoas, no sentido de realizar as atividades do processo e atingir os objetivos do projeto ([BECK 2004]

BECK, Kent; ANDRES, Cynthia. **Extreme Programming Explained:** Embrace Change. 2. ed. Addison-Wesley, 2004.

[GIMENES 1994]).

Desenvolver um novo produto de software é arrisco. Utilizar um processo de software insuficientemente testado intensifica estes riscos. Um processo de software precisa ser estável, conter práticas comprovadas pela indústria, envolver todos os participantes da produção e ser flexível a ponto de permitir configurações e adaptações, a fim de atender às necessidades de produção dos diferentes tipos de produtos, e de equipes e organizações de diferentes tamanhos e natureza.

2.2.2 Modelos Atuais

Atualmente existem vários processos de software. Dentre os que possuem grande destaque no mercado está o Processo Unificado da Rational (RUP) ([LARMAN 2001]). O RUP foi concebido ao longo de vários anos de trabalho e contribuições.

Os três maiores colaboradores do RUP e da Linguagem Unificada de Modelagem (UML) são os “três amigos” – Ivar Jacobson, Grady Booch e James Rumbaugh. Eles são conhecidos na engenharia de software como criadores de uma das mais importantes e reconhecidas empresas de desenvolvimento de software no mundo: a Rational Software Corp. (hoje propriedade da IBM).

O termo "Unificado" da sigla RUP significa a incorporação de inúmeras experiências de profissionais e companhias que contribuíram para o amadurecimento da UML, hoje linguagem padrão para a documentação de todas as etapas do processo de software. Profissionais da Rational Software Corp. também forneceram importantes contribuições.

A disciplina de desenvolvimento de software *Extreme programming* (XP) ([BECK 2004]) vem adquirindo cada vez mais adeptos. Há práticas e métodos de XP que podem ser eficientemente adotados pelo RUP, adicionando recursos que podem vir a melhorar a produção do software ([KROLL & KRUCHTEN 2003], [LARMAN 2003], [LARMAN 2001] e [POLLICE et. al. 2003]).

2.3 Processo Unificado da Rational

O RUP é um produto da IBM Rational. Foi criado com base nos trabalhos dos três fundadores da antiga Rational Software Corp., os "três amigos" – Ivar Jacobson, Grady Booch e James Rumbaugh – antes de sua aquisição pela IBM. Historicamente, o RUP é descrito como o resultado da aplicação da "abordagem Rational" sobre o Processo Baseado em Objetos (*Objectory Process*, [JACOBSON et. al. 1999]).

Além dos três nomes da Rational, inúmeros trabalhos contribuíram para o aprimoramento e evolução do RUP. Este trabalho conjunto é certamente uma das principais razões para seu sucesso e popularidade, que desta forma, integra uma série de técnicas e práticas já utilizadas na indústria e discutidas na academia. O RUP pode ser descrito como uma agregação de várias boas idéias sobre o modo como produtos de softwares são desenvolvidos. Na verdade, pode-se dizer que todos os processos populares, o que inclui *Extreme programming* ([LARMAN 2003]) e *Feature-Driven Development* (Desenvolvimento Baseado em Recursos - FDD) ([LARMAN 2003]), constituem um conjunto de elementos e boas práticas observadas na indústria e na academia. As diferenças entre os processos estão na forma como cada um descreve a aplicação destas práticas e elementos.

Conceitualmente, o RUP é descrito como um *framework* (arcabouço) de processo de software. Adota-se o princípio de que não existe nenhum processo pronto e "empacotado" que possa ser utilizado como diretriz para desenvolver todos os tipos de produto de software. Um processo de software mais amplo, conceituado como uma coleção de componentes de

processo, pode ser configurado (ou ajustado) de forma que seja melhor adequado à realidade do projeto ou domínio no qual será aplicado. Segundo [ROYCE 1998], o RUP pode ser descrito como arquitetura de processo, sendo instanciada para processos específicos com o objetivo de atender às necessidades de diferentes tipos e tamanhos de projetos.

O objetivo do RUP é oferecer uma abordagem disciplinada para definir tarefas e responsabilidades dentro de uma organização de desenvolvimento de software, assim como um conjunto de boas práticas utilizadas na indústria. Considera-se importante assegurar que todos os membros da equipe compartilhem a mesma linguagem, processo e a visão de como desenvolver software. Para tanto, as atividades do RUP são centradas em criar e manter artefatos, utilizando a UML ([JACOBSON et. al. 1999]) como padrão preferencial.

2.3.1 Princípios do RUP

O Processo Unificado da Rational possui três princípios elementares: orientado a casos de uso, centrado em arquitetura e iterativo e incremental. Juntamente com o "Espírito do RUP", apresentado na seção 2.3.2, esses princípios ajudam a definir a forma pela qual o RUP deve ser compreendido e aplicado, assim como, os principais valores de sua aplicação.

2.3.1.1 Orientado a Casos de Uso

Um sistema de software é construído para servir aos seus usuários. Desta forma, para se construir um software de sucesso é preciso conhecer suas necessidades e desejos. O termo usuários não se refere somente aos humanos, mas também a sistemas externos relacionados. No RUP, uma interação é chamada de Caso de Uso (*Use Case*) do sistema e representa uma funcionalidade significativa. A união dos casos de uso forma o Modelo de Casos de Uso, que descreve as funcionalidades do sistema. Esse modelo veio em substituição à tradicional Especificação Funcional de um sistema.

A especificação funcional do sistema é escrita em resposta a uma pergunta: "O que o sistema deve fazer?". A estratégia dos casos de uso pode ser caracterizada pela adição de

outra pergunta pergunta: “Para qual usuário?”. Desta forma, os engenheiros analisam o valor que o software tem para seus usuários e não somente as funções que podem ser úteis ao processo de negócio da empresa. Contudo, os casos de uso não são somente um mecanismo para especificação dos requisitos de um sistema, eles também orientam seu projeto, implementação e teste. Por esta razão, o RUP é dito um processo orientado a casos de uso.

O conceito de ser orientado por caso de uso significa que o fluxo do processo evolui a partir da especificação destes. Os casos de uso são especificados, usados como base para análise e projeto, implementados e, tornam-se referência para realização dos testes do sistema ([KRUCHTEN 2000]).

2.3.1.2 Centrado em Arquitetura

Segundo [JACOBSON et. al. 1999], os casos de uso e a arquitetura estão relacionados, pois todo software precisa ter tanto função, quanto forma. Esses dois aspectos precisam ser balanceados para obter um produto de software de sucesso. No RUP, as funções do software correspondem aos casos de uso, e a forma, à arquitetura. Ambos precisam evoluir em paralelo: a arquitetura deve ser construída em uma seqüência lógica, que está diretamente relacionada com o levantamento, detalhamento e evolução dos casos de uso do sistema:

- Na primeira etapa do processo, deve-se criar uma versão mais rústica da arquitetura para incorporar requisitos não funcionais, como por exemplo, a plataforma na qual o software vai funcionar. Em paralelo, os casos de uso do sistema são identificados e descritos de forma sucinta.
- A seguir, devem-se priorizar as funcionalidades mais significativas do sistema, que possuem uma importância estrutural (arquitetural). Desta forma, trabalha-se com um subconjunto de casos de uso que representam as funcionalidades principais do sistema (5% a 10% do conjunto completo). Cada caso de uso é selecionado, especificado em detalhes e abstraído em termos de subsistemas, classes e

componentes.

- À medida que os casos de uso são especificados e detalhados, outras partes da arquitetura acabam sendo definidas. Trabalhando-se em etapas (iterações), chega-se a versões mais estáveis do software, sempre priorizando o detalhamento e a implementação dos casos de uso mais significativos para o sistema.

Estes três passos são repetidos iterativamente até que a arquitetura esteja em uma versão estável. Desta forma, a construção do software é guiada pelos casos de uso e prioriza a arquitetura do sistema.

2.3.1.3 Iterativo e Incremental

Desenvolver um produto de software é um empreendimento de meses ou anos. Uma solução comum é dividir esta tarefa em partes menores ou montagens (*builds*). No RUP, cada um destes mini-projetos é uma iteração que resulta em um incremento do sistema ([KROLL & KRUCHTEN 2003]) (ver Diagrama 1).

A abordagem iterativa e incremental é talvez a mais importante característica do RUP. O desenvolvimento é organizado em uma série de iterações curtas (por exemplo, quatro semanas). Ao final de cada uma, o software produzido é testado e integrado, e tem-se uma versão executável interna (lançamento ou *release* interno). Cada iteração inclui suas próprias atividades de análise de requisitos, projeto, implementação e testes ([KRUCHTEN 2000]). As várias iterações resultam, possivelmente, em um lançamento externo para o cliente.

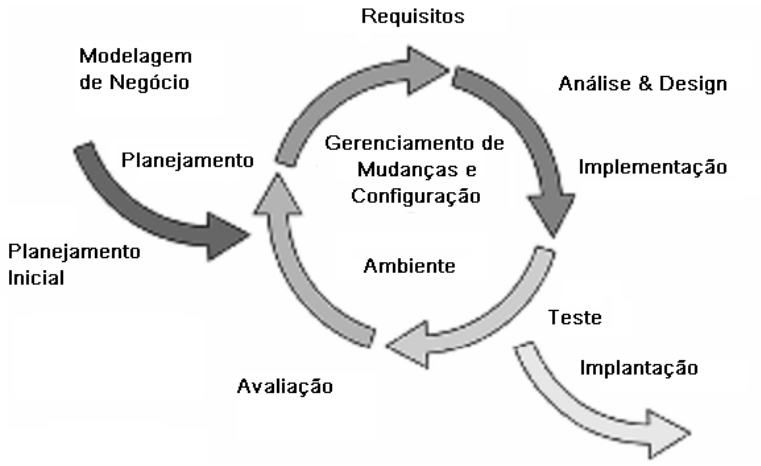

Diagrama 1: Abordagem iterativa e incremental. [KRUCHTEN 2000].

O ciclo de vida iterativo é baseado em uma sucessiva ampliação e refinamento do sistema por meio de múltiplas iterações, e é dirigido por retornos (*feedbacks*) e adaptações, permitindo que modificações e melhorias sejam identificadas e implementadas em iterações posteriores. Essa flexibilidade dá ao time de desenvolvimento maior agilidade para responder às mudanças nos requisitos. Dessa forma, o sistema cresce incrementalmente em iterações.

Segundo [POLLICE et. al. 2003] e [LARMAN 2001], existe uma série de vantagens no uso de um processo iterativo. Dentre elas estão:

- *Detecção antecipada de riscos*. A cada iteração é feita uma avaliação e planejamento sobre o andamento do projeto. Retornos da iteração anterior ajudam a definir estratégias de contenção e mitigação tanto dos riscos já conhecidos como de novos que possam ser detectados no decorrer do projeto.
- *Visualização do progresso*. O modelo iterativo e incremental torna a progressão facilmente mensurável, pois o trabalho é dividido em uma série de mini-projetos que representam a construção de cada parte do sistema.
- *Retorno (retorno) imediato e constante*: Adaptações que precisam ser feitas no projeto são detectadas mais cedo e permitem que mudanças levem o sistema a um estado mais refinado, atendendo melhor as reais necessidades de seus interessados.

- *Gerenciamento da complexidade*. A abordagem iterativa força a divisão dos problemas em problemas menores, diluindo a complexidade. Assim, a equipe de desenvolvimento não fica "emperrada" em problemas de análise ou questões muito complexas (ou longas).
- *Adaptação do processo*. O aprendizado obtido em uma iteração pode ser usado para melhorar o próprio processo de desenvolvimento, refinando-o para melhor atender as necessidades do projeto.

2.3.2 Essência do RUP

O RUP é um *framework* de processo, utiliza a abordagem iterativa e incremental, enfatiza a evolução da arquitetura do sistema o mais cedo possível e baseia o desenvolvimento sobre os casos de uso do sistema. Existe um conjunto de práticas que representa o chamado "Espírito" (ou essência) do Processo Unificado da Rational. Segundo [POLLICE et. al. 2003], [KROLL & KRUCHTEN 2003], [KRUCHTEN 2000], o espírito do RUP representa seu conceito fundamental: se estas práticas não estão sendo seguidas, não é correto caracterizar como adoção do RUP. Essas práticas compreendem:

- *Atacar riscos antes e continuamente*. Devem-se concentrar esforços para mitigar os riscos do projeto assim que forem identificados, independentemente de sua natureza (negócios, técnicos, etc). O "ataque" aos riscos deve ser constante: no começo e durante cada iteração do projeto.
- *Assegurar entrega de valor ao cliente*. O gerenciamento de requisitos e outras atividades de engenharia são essenciais para o projeto, mas deve-se assegurar que os requisitos do sistema sejam concretizados, entregando valor real ao cliente: software de qualidade em funcionamento.
- *Utilizar software executável como indicativo de progresso*. Documentos, projetos e planos são indispensáveis ao projeto, mas são péssimos indicadores de progressão

para com o cliente, pois possuem pouco valor para ele quando comparados ao software em funcionamento. Código que compila e passa pelos testes é o melhor indicador de progresso no projeto.

- *Possibilitar mudanças.* As aplicações atuais são muito complexas para que se consiga desvendar todos os seus problemas em uma única etapa. Desvendar as soluções de grandes problemas é uma tarefa melhor realizada de forma incremental. Os requisitos, projeto e implementação do software constituem atividades que dificilmente são desenvolvidas de forma 100% correta de forma seqüencial. Mudanças vão ocorrer cedo ou tarde, pois elas são indicativas de melhorias necessárias, e o projeto deve permitir que elas sejam executadas.
- *Priorizar e antecipar implementação da arquitetura.* Toda solução possui um alicerce, que dita a forma como todas as outras partes complementares da solução serão implementadas. Devem-se priorizar esforços para definir e construir a arquitetura do sistema. Desta forma, provas de conceito e mudanças críticas, que geralmente impactam sobre o alicerce do software, podem ser feitas no início do projeto, eliminando vários riscos que seriam caudados pelo impacto destas mudanças sendo implementadas em fases posteriores.
- *Priorizar a componentização do sistema.* Problemas complexos ou grandes são melhor compreendidos e solucionados se trabalhados de forma modular. O paradigma orientado a objetos permite que aplicações sejam mais adaptáveis a mudanças à medida que as soluções são modularmente encapsuladas em componentes do sistema. Além disso, componentes melhoram o reutilização e diminuem a propagação de erros, uma vez que são manipulados por meio de *interfaces*.
- *Trabalhar como um time.* O desenvolvimento de software é um trabalho realizado em equipe. A abordagem iterativa enfatiza a importância de uma boa comunicação

e relacionamento entre pessoas, além de permitir que cada integrante possa participar de forma mais ativa do desenvolvimento do software. Isso aumenta o sentido de responsabilidade de cada um sobre o resultado da equipe.

- *Priorizar a qualidade.* Testes não podem ser realizados somente do final do desenvolvimento de software. Devem-se fazer testes antes, durante e depois da implementação. Diferentes tipos de testes podem ser aplicados ao software, de acordo com a sua natureza. No entanto, qualidade envolve mais do que testes. Ela deve estar na execução do trabalho diário, partindo das pequenas tarefas, de todos os integrantes da equipe. Qualidade deve ser vista como uma filosofia e não como uma característica do processo ou exigência organizacional.

2.3.3 Estruturas Dinâmica e Estática do Processo

O Processo Unificado da Rational possui uma arquitetura que pode ser descrita em termos de duas estruturas (ou dimensões), mostrada na Figura 1.

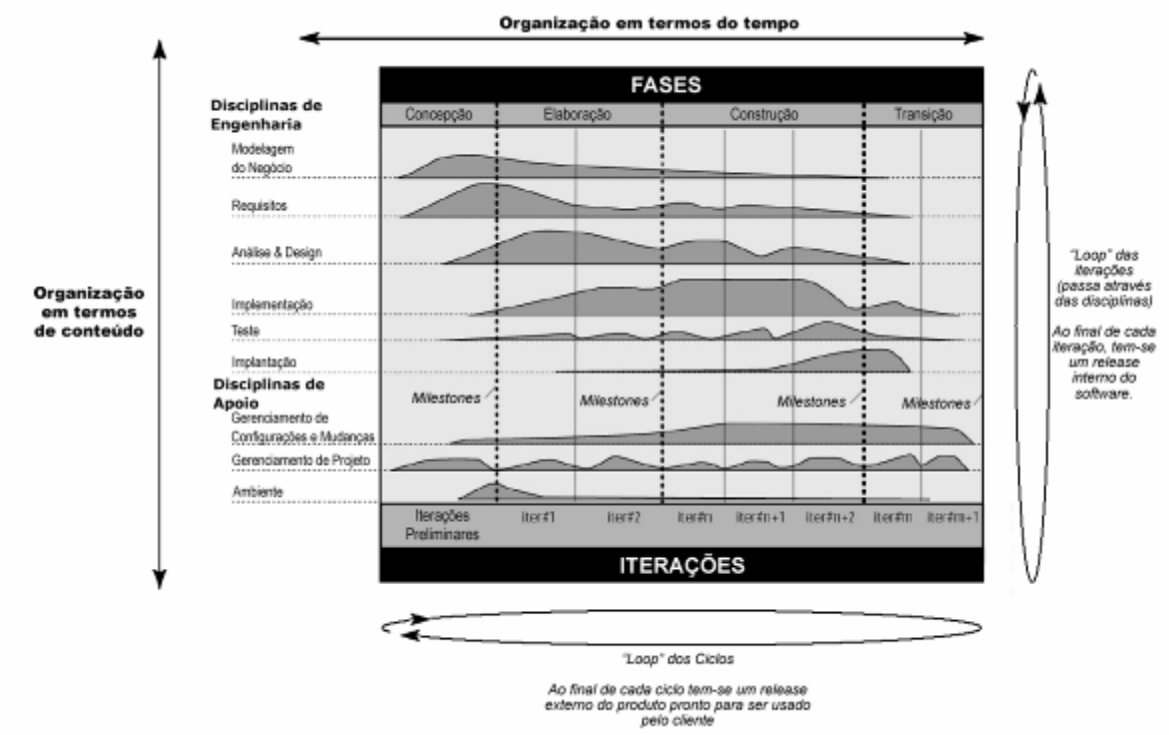

Figura 1: Estrutura em duas dimensões [KRUCHTEN 2000].

- *Estrutura Horizontal* – Representa a organização do processo em termos de tempo e mostra os aspectos dinâmicos do ciclo de vida do processo, sendo expressa em ciclos, fases e Marcos Principais. O processo é dividido em quatro fases seqüenciais e a mudança de uma fase para outra ocorre quando se alcança um marco principal (conjunto de metas) previamente definido (ver Figura 2). São elas: Concepção (*Inception*), Elaboração (*Elaboration*), Construção (*Construction*) e Transição (*Transition*). O conjunto formado por estas quatro fases constitui um ciclo de desenvolvimento. Ao final de cada ciclo, tem-se um lançamento externo. Um projeto pode ser executado em vários ciclos, dependendo do número de lançamentos externos.
- *Estrutura Vertical* - Representa o aspecto estático do processo, sendo descrito em termos de atividades, artefatos, papéis e disciplinas. Cada disciplina possui um conjunto de atividades relacionadas, que por sua vez, produzem artefatos específicos. O RUP é configurado para um domínio específico em termos das atividades, artefatos e papéis que serão necessários para atender às suas necessidades. É na estrutura vertical que ocorrem as iterações, passando por meio das atividades das disciplinas do processo. A quantidade de esforço aplicado em cada uma das atividades das disciplinas muda de acordo com a progressão através das quatro fases do RUP (ver Figura 1). Desta forma, há convergência para um refinamento dos requisitos, retornos, construção incremental e adequação do processo. Ao final de cada iteração, tem-se um lançamento interno. Esta é uma organização do processo em termos de conteúdo.

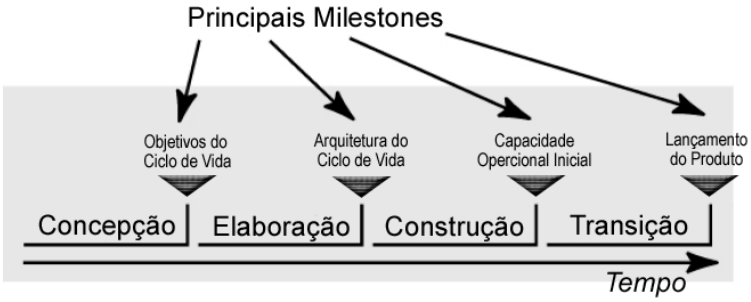

Figura 2: Fases e Marcos principais do processo [KRUCHTEN 2000].

2.3.4 Estrutura Dinâmica do Processo

Divide o ciclo de desenvolvimento em quatro fases sucessivas, por meio das quais os artefatos são produzidos. Cada uma é concluída quando se alcança um marco principal - um ponto onde são alcançadas metas-chave. Quando todas as iterações planejadas para a fase são concluídas, critérios de avaliação são aplicados pra detectar se o marco principal foi alcançado. Em caso negativo, uma ou mais iterações são executadas solucionar as pendências. O conceito essencial de cada fase do RUP é apresentado abaixo.

2.3.4.1 Concepção

O propósito maior da fase de concepção é compreender o sistema, formulando uma concepção em alto-nível de seus requisitos e estabelecendo seu escopo. Analisam-se os riscos associados ao projeto, realiza-se o planejamento do projeto e obtêm-se aprovação dos interessados para prosseguir.

Objetivos

- Compreender o escopo do projeto.
- Construir o artefato Plano de Desenvolvimento (configuração do processo).
- Obter a aprovação dos interessados para prosseguir com o projeto.

Critérios de Avaliação para progressão à próxima fase

- Consentimento dos interessados com respeito à definição do contexto e estimativas de custo e tempo (cronograma).
- Entendimento dos requisitos, o que é comprovado pela fidelidade dos primeiros casos de uso.
- Credibilidade das estimativas de custo e tempo (cronograma), prioridades, riscos, e processo de desenvolvimento.
- Desenvolvimento do primeiro protótipo da arquitetura.
- Avaliação das despesas reais contra despesas planejadas.

Resultados esperados da fase de Concepção

- Um documento de visão: uma visão geral das exigências centrais do projeto, características chaves, e questões principais.
- Um modelo de casos de uso inicial, com uma descrição breve de todos os casos de uso identificados até o momento. Detalhar os arquiteturalmente significativos (10% a 20% do total).
- Um glossário de projeto inicial (pode ser expresso como um modelo de domínio).

- Um caso de negócio inicial, que inclui o contexto empresarial, critérios de sucesso (projeção de renda, reconhecimento de mercado, etc.), e previsão financeira.
- Uma análise de risco inicial.
- Um plano de projeto, mostrando fases e iterações.
- Um modelo empresarial, se necessário.
- Um ou vários protótipos.

Marco Principal de conclusão da fase de Concepção

- Objetivos do Ciclo de Vida devem estar bem definidos (*Lifecicle Objectives Milestone - LCO*).

2.3.4.2 Elaboração

Nesta fase, existe uma maior concentração de esforços em tarefas técnicas: projeto, implementação, testes, e produção da arquitetura executável do software. Ao final da fase a arquitetura deve incluir subsistemas, interfaces, componentes chave e mecanismos como comunicação entre processos e persistência de dados. A solução de riscos técnicos de maior periculosidade deve ser priorizada, tais como sincronia e acesso a recursos, performance e segurança de dados.

Objetivos

- Mitigar riscos técnicos de maior impacto.
- Criar uma arquitetura executável estável.
- Conseguir uma visão clara do que é necessário para se construir o sistema.

Critérios de avaliação da fase de Elaboração

- A visão do produto está estável?
- A arquitetura está estável?
- O software executável (demonstrativo) mostra que os elementos de risco principais foram identificados e solucionados de forma confiável?
- O plano para a fase de construção está suficientemente detalhado e preciso? Está apoiado em uma base confiável de estimativas?
- Todos os interessados no sistema concordam que a visão atual pode ser alcançada se o plano atual for executado para desenvolver o sistema completo, no contexto da arquitetura atual?
- O planejamento dos custos está de acordo com os custos reais?

Resultados esperados da fase de Elaboração

- Um modelo de casos de uso (pelo menos 80% completo) onde foram identificados todos os casos de uso e atores, e a maioria das descrições destes casos foi desenvolvida.

- Requisitos adicionais que capturam as exigências não-funcionais e quaisquer outras que não são associados com nenhum caso de uso específico.
- Uma Descrição da Arquitetura do Software.
- Um protótipo de arquitetura executável.
- Uma lista de riscos revisada e um plano empresarial (*business case*) revisado.
- Um plano de desenvolvimento para o projeto todo, incluindo o plano de projeto detalhado, que mostra as iterações e os critérios de avaliação para cada uma delas.
- Um Plano de Desenvolvimento atualizado que especifica o processo a ser usado.
- Um manual de usuário preliminar (opcional).

Marco Principal de conclusão da fase de Elaboração

- Arquitetura Executável do software deve estar bem definida e funcional (*Lifecicle Architecture Milestone - LCO*).

2.3.4.3 Construção

Nesta fase é realizada a maior parte da implementação do sistema, progredindo da arquitetura executável para a primeira versão operacional do sistema. Várias iterações são executadas e resultam em lançamentos internos e externos (também chamados de lançamentos Alpha). As versões Alpha são passadas aos interessados para que seus retornos possam ajudar a garantir que o sistema está funcional e satisfaz suas necessidades. Ao final da fase, tem-se uma versão Beta do sistema. Esta versão é considerada completa e ainda instável: é totalmente funcional, incluindo mecanismos de instalação, documentação e material de treinamento. Neste ponto ainda não se tem o sistema por finalizado: são necessárias ajustes de funcionalidades, performance e qualidade.

Objetivos

- Construir a primeira versão operacional do software.

Critérios de avaliação da fase de Construção

- O lançamento do produto está estável e maduro o bastante para ser implementado no ambiente do usuário?
- Todos os interessados estão prontos para a transição no ambiente do usuário?
- As despesas de recursos atuais estão de acordo com as despesas planejadas?

Resultados da fase de Construção

- O produto de software integrado nas plataformas adequadas.
- Os manuais de usuário.
- Uma descrição do lançamento atual.

Marco Principal de conclusão da fase de Construção

- Deve estar disponível uma primeira versão operacional do sistema (*Initial Operational Capability Milestone - IOC*).

2.3.4.4 Transição

O objetivo maior da fase de transição é assegurar a máxima satisfação de seus interessados através de pequenos ajustes finais. São realizados testes e pequenas modificações baseadas em retornos. Neste ponto do ciclo de vida, os retornos dos usuários devem estar centrados em ajustes, configuração, instalação e detalhes de usabilidade. Questões estruturais devem ter sido resolvidas em fases e iterações anteriores. Por esta razão é que retornos contínuos dos interessados durante todo o ciclo de vida são essenciais para produzir software de qualidade.

Objetivos

- Construir a versão final do produto e entregá-la.

Critérios de avaliação da fase de Transição

- O usuário final está satisfeito?
- As despesas de recursos atuais estão ainda aceitáveis, contra as despesas planejadas?

Marco Principal de conclusão da fase de Transição

- Deve estar disponível e entregue a versão final do sistema (*Product Release Milestone - PR*).

2.3.5 Estrutura Estática do Processo

A estrutura estática do RUP divide o processo em quatro elementos: Papéis, Atividades, Artefatos e Disciplinas (ou *Workflows*). Segundo [JACOBSON et. al. 1999], um processo descreve "quem" está fazendo "o que", "como" está sendo feito e "quando". No RUP, os Papéis (“quem”) executam Atividades (“como”) para produzir Artefatos (“o que”) ao longo de quatro Fases (“quando”) do ciclo de vida do software. Os papéis, atividades e

artefatos do RUP estão agrupados em Disciplinas.

Algumas práticas e princípios do RUP não são configuráveis, tais como o desenvolvimento iterativo, o gerenciamento dos riscos e a verificação contínua da qualidade. No entanto, uma característica-chave e fundamental do RUP é que todas as atividades e artefatos (modelos, diagramas, documentos, etc) são opcionais, com exceção óbvia do código. O numeroso conjunto de papéis, atividades, disciplinas e artefatos descritos no RUP devem ser filtrados para aquele que realmente resolve as necessidades do projeto. Os artefatos são exemplos comuns: não devem ser produzidos aqueles que não possuam utilidade para o trabalho, pois isto seria desperdício de tempo, esforço e investimentos. Segundo [POLLICE et. al. 2003], é aconselhável que uma equipe direcione o foco para um pequeno conjunto de artefatos que seja de grande valia para o seu trabalho.

O conjunto de artefatos selecionados para um projeto é comumente descrito no Plano de Desenvolvimento (*Development Case*), artefato da disciplina Ambiente.

2.3.5.1 Papéis

Um papel define o comportamento e responsabilidades de uma pessoa ou grupo. Um indivíduo pode desempenhar vários papéis. As responsabilidades que são atribuídas a um papel incluem tanto executar certo conjunto de atividades como ser o mantenedor de um conjunto de artefatos.

2.3.5.2 Atividades

Uma atividade é um trabalho que um indivíduo com determinado papel, pode ser solicitado a executar. A atividade tem um propósito claro, normalmente expresso em termos de criar ou atualizar alguns artefatos, tais como um modelo, classe ou plano. Toda atividade é nomeada a um papel específico, e geralmente são definidas para serem desempenhadas em horas ou dias, afetando um ou mais artefatos. Uma atividade deve ser planejada para que haja progresso: se é muito pequena, será negligenciada, e se é muito grande, é preciso expressar o

progresso em termos de partes desta atividade ([KROLL & KRUCHTEN 2003]).

Alguns exemplos de atividades são: Planejar uma iteração, para o Papel Gerente de Projeto; Descobrir Casos de Uso e Atores, para o Papel Analista de Sistemas; Revisar o Projeto, para o Papel Revisor de Projeto, e; Executar Testes de Desempenho, para o Papel Testador de Desempenho.

2.3.5.3 Artefatos

Os artefatos são os produtos tangíveis do projeto, ou em outras palavras, as coisas que o projeto produz, modifica ou usa enquanto uma atividade é executada. Eles são usados tanto como entradas para uma tarefa, quanto como resultado ou saída desta. Eles podem ter vários formatos: modelos, como o Modelo de Casos de Uso ou o Modelo de Projeto; elementos de modelo, tal como uma classe, um caso de uso ou um subsistema; documentos, como o Caso de Negócio ou o Documento de Arquitetura do Software; códigos fonte; ou mesmo, executáveis do software.

2.3.5.4 Disciplinas ou Workflows

O conjunto de todos os papéis, atividades e artefatos não constituem por si só um processo. É preciso um modo para descrever as seqüências de atividades que produzem algum resultado, assim como para mostrar interações entre papéis. Uma disciplina é uma sucessão de atividades que produzem um resultado de valor concreto (em outras palavras, um artefato). Em uma interação, passa-se pelas disciplinas do RUP, executando as atividades de acordo. Existem nove disciplinas no Processo Unificado da Rational.

2.3.6 Disciplinas do RUP

Existem nove disciplinas no RUP, que agrupam os papéis e atividades de forma lógica. Como mostrado na Figura 3, elas são divididas em dois grupos: Disciplinas de Engenharia e Disciplinas de Apoio.

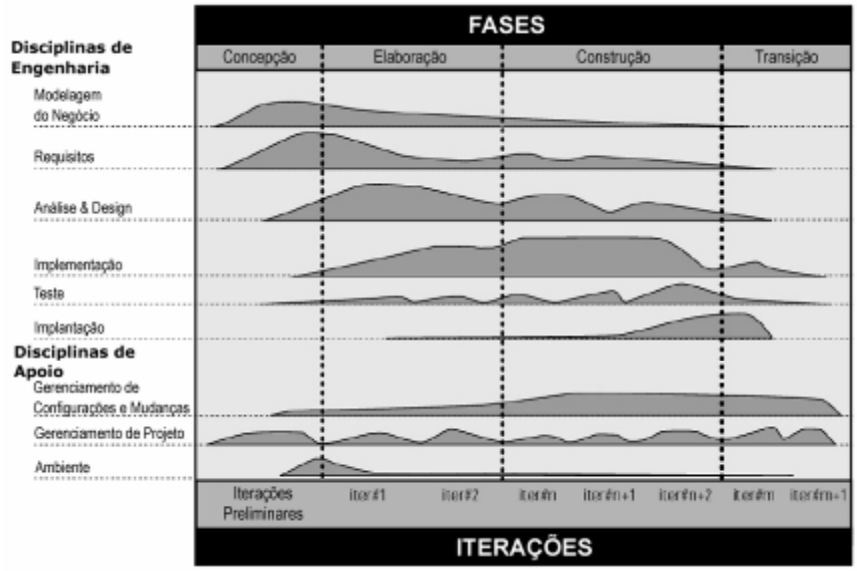

Figura 3: Disciplinas e distribuição de esforço [KRUCHTEN 2000].

Embora os nomes das seis disciplinas fundamentais de engenharia possam lembrar as fases seqüenciais de um processo tradicional em cascata, as fases do RUP são outras (seção "Estrutura Dinâmica do Processo"). Essas disciplinas são agrupamentos lógicos de atividades relacionadas, e conseqüentemente, de Papéis e Artefatos. O processo intercala estas nove disciplinas e as repete com diferentes ênfases e intensidade a cada iteração (Figura 3). Segundo [KRUCHTEN 2000], isto converge para um refinamento dos requisitos do sistema, retorno dos interessados, construção incremental do software e adaptações para adequação do processo.

2.3.6.1 Modelagem do Negócio

A disciplina de Modelagem do Negócio *(Business Modeling*) engloba as atividades necessárias que tem como objetivo identificar, compreender e documentar os processos da empresa. Geralmente estes são documentos no modelo de Casos de Uso do Negócio (*Business Use Cases*). O objetivo desta disciplina é assegurar um entendimento comum entre todos os interessados a respeito de qual processo empresarial precisa receber apoio na organização.

Os casos de uso do negócio são analisados para se entender como o negócio deve apoiar os processos empresariais. Muitos projetos podem não utilizar as atividades de

modelagem do negócio no seu processo, principalmente nos casos em que o domínio do negócio já é muito bem conhecido por todos os participantes do projeto (principalmente os engenheiros) ou então, quando o negócio é simples e pequeno.

2.3.6.2 Requisitos

As atividades desta disciplina são realizadas a fim de se descrever o que o sistema deve fazer, assim como permitir aos desenvolvedores e clientes avaliarem esta descrição. Tais atividades englobam identificar, organizar, e documentar as funcionalidades exigidas, assim como rastrear e documentar mudanças e decisões. Assim como a Modelagem do Negócio, existe maior concentração de esforços para a realização das atividades inerentes a esta disciplina nas duas primeiras fases do processo, Concepção e Elaboração.

Um documento de Visão é criado e a partir deste, são definidas as necessidades dos interessados no sistema e identificados os atores. Ao longo do projeto, cada caso de uso é descrito em detalhes, mostrando interação com atores e suas funcionalidades, assim como os fluxos das regras do negócio. São descritas exigências não-funcionais em especificações adicionais. Os casos de uso funcionam como guia ao longo do ciclo de desenvolvimento do produto de software.

2.3.6.3 Análise & Projeto

As atividades da disciplina de Análise & Projeto são executadas para demonstrar como o sistema será concebido na fase de Implementação. O projeto de um sistema deve garantir que o sistema a ser construído execute as tarefas e funções especificadas nos casos de uso, contemple requisitos não-funcionais, e seja flexível às mudanças. As atividades desta disciplina resultam em um Modelo de Projeto e opcionalmente em um Modelo de Análise. O modelo de projeto serve como uma abstração do código fonte: um guia ou mapa de como o código deve ser estruturado e escrito. Todas as soluções de análise e projeto do sistema devem levar em conta a arquitetura do sistema, foco principal das primeiras iterações do ciclo de vida

do sistema.

2.3.6.4 Implementação

As atividades desta disciplina compreendem definir a organização do código em subsistemas, implementar e testar componentes, e integrar os resultados, formando um sistema executável. O RUP propõe a prática de reutilização de componentes existentes, acelerando a produtividade. Componentes são estruturados em subsistemas da implementação, podendo ser heterogêneos (produzidos com diferentes tecnologias).

2.3.6.5 Teste

Na disciplina de Testes encontram-se as atividades responsáveis por verificar as funcionalidades do sistema, a integração de objetos e componentes, a validade dos requisitos implementados, assim como, assegurarem que os defeitos encontrados sejam solucionados. De acordo com a abordagem iterativa proposta pelo RUP, os testes são realizados ao longo de todo ciclo de vida do software, e não apenas no final. Isto permite encontrar defeitos tão logo quanto possível, reduzindo custos de eventuais correções posteriores.

No Processo Unificado da Rational, os testes são executados considerando as seguintes dimensões de qualidade: confiabilidade, funcionalidade, e desempenho da aplicação e do sistema ([JACOBSON et. al. 1999]). Uma prática importante quando se usa uma abordagem iterativa é a automatização de testes, pois permite regressão ao fim de cada iteração, assim como, validar cada nova versão do produto por meio de critérios estabelecidos.

2.3.6.6 Implantação

A disciplina de Implantação agrupa as atividades relacionadas ao lançamento e implantação do software, tais como: produzir lançamentos externos (para o cliente); "empacotar", distribuir e instalar o software, e; oferecer ajuda e assistência aos usuários. Em muitos casos, a disciplina também inclui atividades como planejar e gerenciar testes *beta*;

migração de software ou dados existentes, e; condução à aceitação formal do software pelos interessados ([KRUCHTEN 2000]).

2.3.6.7 Gerenciamento de Projeto

O Gerenciamento de Projetos inclui atividades como balancear objetivos, gerenciar riscos e contornar problemas a fim de que se possa entregar um software com sucesso. O Processo Unificado da Rational oferece alguns recursos para a realização das atividades: um *framework* de Gerência de Projetos, com diretrizes para planejar, alocar pessoal, executar e monitorar projetos, e; um *framework* de Gerência de Riscos. A disciplina não é uma receita para sucesso, mas apresenta uma abordagem que pode melhorar a tarefa ([KRUCHTEN 2000]).

2.3.6.8 Gerenciamento de Configurações e Mudanças e Ambiente

Esta disciplina engloba o controle dos diferentes artefatos produzidos em um projeto. Em conjunto com a disciplina de Ambiente, ela contém as atividades referentes ao gerenciamento de versão e configuração do software: base de conhecimentos, repositórios de dados e informações, e controle de mudanças do projeto.

2.4 Sumário

O Processo Unificado da Rational é considerado um processo consolidado pela indústria e passível de ser aplicado a diferentes domínios de aplicações através de seu mecanismo de configuração (ou *tailoring*). O objetivo deste capítulo esteve em apresentar o RUP de forma compatível ao escopo desta discussão. Um dos pontos favoráveis do RUP para o desenvolvimento de jogos é a possibilidade de adotá-lo de forma mais ágil (seção 2.3.2), mesmo considerado por alguns como um processo extenso e rígido.

Como *framework* de processo, o RUP deve ser configurado de maneira a satisfazer as necessidades do domínio ao qual estará sendo aplicado. Isso não é diferente quando se considera a aplicação no desenvolvimento de jogos.

Em continuidade ao estudo proposto, o próximo capítulo apresenta uma visão geral sobre o desenvolvimento de jogos.

3 Desenvolvimento de Jogos

3.1 Introdução

O processo de desenvolvimento de jogos tem evoluído desde seu início. Os primeiros jogos eram desenvolvidos por equipes pequenas e não existia a necessidade de usar um processo formal de produção. Com o tempo, os jogos se tornaram mais complexos, as equipes cresceram e a necessidade por um controle maior foi emergindo naturalmente. Atualmente, o Processo Unificado da Rational é um dos principais processos usados na área ([GAMASUTRA 2005]e [GAMEDEV 2005]).

O processo de desenvolvimento e o processo de criação de jogos (*game design*) são discutidos neste capítulo. Os papéis, artefatos e etapas de produção de jogos são apresentados de forma resumida e objetiva. Ao final, as tecnologias relacionadas à produção de jogos móveis são listadas.

3.2 Jogos: Cinematografia e Interatividade

Desde a renovação dos videogames na década de 80 com o surgimento dos consoles da chamada terceira geração, representados pelos Nintendo NES/Famicom e Sega Master System, o mercado de jogos tem crescido de forma contínua e gigantesca. Hoje é um dos mercados de entretenimento doméstico com maior faturamento anual - sete bilhões de dólares em 2004 -, chegando cada vez mais perto do mercado cinematográfico, que fechou o ano de 2004 com um faturamento de nove bilhões de dólares [ESA 2005].

Há quase duas décadas, os jogos eletrônicos eram tidos pela maioria das pessoas apenas como um passatempo ocasional. Hoje, eles são a principal forma de entretenimento caseiro e parte da vida de muitas pessoas [ESA 2005]. No entanto, o mercado de jogos não é indivisível, existindo diferentes nichos com características bastante específicas: *videogames*,

PC-games, web games, mobile games, advergames, etc.

Jogos possuem muitas semelhanças com filmes: estória, roteiro, personagens, cenários, músicas e efeitos sonoros. Todos esses elementos são necessários para criar um mundo virtual onde o jogador interage com imersão. Pode-se dizer que um jogo é um filme interativo no qual a ação acontece em tempo-real e o jogador está no papel do personagem principal da trama. Em filmes, cada pequeno detalhe da ação já está pré-estabelecido e o usuário é um telespectador.

Nos dias atuais, os jogos usam a cinematografia como elemento chave para a imersão do jogador no seu ambiente virtual e como agente captador de usuários. Entende-se como cinematografia o conjunto de elementos que trazem ao jogo uma riqueza de detalhes que geralmente só é encontrada em filmes. Exemplos desses elementos são: vídeos filmados, animados ou gerados em estações gráficas; trilha sonora produzida em estúdio, arranjada em instrumentos; efeitos sonoros, vozes e narrações; e, personagens com movimentação realista (utilizando ou não mecanismos de captura de movimentos). Pode-se dizer que a cinematografia é responsável por acrescentar uma qualidade exuberante aos videogames.

A cinematografia um dos fatores responsáveis por aumentar a complexidade do projeto de um jogo. Outro fator é o *gameplay* (jogabilidade). Quanto mais extensa e realista é a liberdade de interação com os elementos do jogo, maior é o número de possibilidades e a complexidade do *gameplay*. Ela pode ser descrita como a forma por meio da qual o jogador interage com o jogo e seu mundo virtual, consistindo de um elemento primordial para os videogames ([BETHKE 2003]).

Atualmente, a complexidade de projetos de jogos demanda maiores esforços e investimentos. Muitos jogos possuem orçamentos acima de filmes e contam com uma equipe bastante numerosa para garantir a cinematografia da estória, a imersão do jogador e um *gameplay* mais profundo. A produção de um jogo envolve a formação de uma equipe bastante multidisciplinar, incluindo artistas gráficos, sonoplastas, roteiristas, desenvolvedores de

software e os chamados criadores de jogos (*game designer*). Esta é uma das características que distingue bastante a produção de jogos de outros tipos de softwares ([BETHKE 2003]).

3.3 Game Design

A Criação ou Concepção de Jogos (*Game Design*) é uma área de estudo e uma profissão, podendo ser descrita como a arte de criar jogos ([BETHKE 2003]). Assim como a produção de software, ela envolve técnicas, metodologias, práticas e criatividade. Pode-se dizer que todo processo de criação é naturalmente iterativo e incremental: cria-se uma versão inicial, avaliam-se as boas e más características (retornos), definem-se os próximos passos e inicia-se a próxima etapa do trabalho. Age-se desta forma gradualmente, repetindo os mesmos passos até que se chegue à versão final. Um esquema deste processo é apresentado no Diagrama 2. Os jogos funcionam da mesma maneira sendo que sempre há um planejamento inicial. Ao conjunto de elementos-chave do *gameplay* dá-se o nome de *core gameplay* [PEDERSEN 2003].

Antigamente conhecida como *game planning*, a criação de jogo descreve todas as atividades relacionadas à concepção de um jogo, envolvendo um planejamento completo e detalhado. Este planejamento é realizado antes, durante (e até após) a produção do jogo.

O processo de criação envolve a capacidade de visualizar o jogo por completo no início de sua concepção, sem nenhuma parte funcional implementada. É de responsabilidade do Criador de Jogos definir todos os pontos de um jogo: os objetivos e recompensas, assim como as regras e procedimentos necessários para se completar tais objetivos; enredo e dramaticidade; personagens; ambientes; diálogos, etc.

O Criador de Jogos é o responsável por planejar tudo o que for necessário para que se possa fazer do jogo uma experiência interativa, interessante ao jogador. Considerado o idealizador do jogo, o Criador de Jogos possui uma tarefa pouco simples [FULLERTON et. al. 2004].

De acordo com [FULLERTON 2004], o Criador de Jogos é o principal responsável pela experiência de se jogar. Desde a concepção até a finalização de um jogo, o Criador de Jogos deve assegurar que o *gameplay* esteja de acordo com a forma planejada sobre todos os detalhes do jogo. Como o *gameplay* está intrinsecamente ligado à forma como o jogo é programado, visualizado e contextualizado por músicas, efeitos e vozes, o Criador de Jogos precisa interagir constantemente com todos os demais membros da equipe de produção.

Desde seu início, a criação de jogos tem evoluído, passando a constituir uma área de estudo. Atualmente, academia e indústria têm despendido esforços a fim de alcançar melhores (e mais padronizadas) formas de se conceber e descrever um jogo.

Erroneamente descrito pela simplicidade de um "fornecedor de idéias", o Criador de jogos precisa ir muito além. Sabe-se que todo jogador é capaz de ter idéias novas e originais sobre jogos. No entanto, uma idéia somente não é suficiente para se criar um jogo: é a sinergia de todos os elementos que o compõe que criam uma experiência de interação interessante ao usuário. Esses elementos precisam ser minuciosamente planejados, detalhados e descritos.

Dependendo da riqueza de detalhes (e do tamanho) do jogo, a tarefa de criação pode se tornar muito complexa e arriscada para uma só pessoa executar. Nestes casos, é constituída uma equipe de criação de jogos. Essa situação é encontrada na grande maioria dos projetos de jogos para computadores pessoais e consoles de videogames atuais, que possuem uma complexidade alta demais para que um só Criador de Jogos possa definir todos os seus detalhes.

A criação de jogos pode ser descrita como um conjunto de atividades, podendo ser executadas por diferentes pessoas ([ROUSE 2000]):

- Criador de Níveis (*Level Designer*): descreve a estrutura de todos os ambientes do jogo. Esta tarefa compreende a criação dos cenários, o posicionamento de cada um de seus elementos - oponentes, obstáculos, itens, objetivos e recompensas, dentre outros.

Atualmente, este é um dos papéis fundamentais na produção de jogos, uma vez que os ambientes constituem o mundo virtual onde o jogo acontece. Este papel atua em conjunto com artistas gráficos para definir da melhor forma os elementos visuais dos cenários.

- Criador de Personagens (*Character Designer*): descreve o conceito de todos os personagens do jogo. Também pode definir comportamentos de personagens não controlados pelo jogador (*NPC's – Non-Player Characters*). Trabalha em conjunto com artistas gráficos para definir a parte visual dos personagens.
- Criador de Mecânicas (*Gameplay Designer*): descreve todas as regras, procedimentos de interação, objetivos e recompensas da interação do usuário com o jogo. É responsável por definir grande parte do *gameplay* do jogo.
- Criador de Roteiro (*Script Designer*): descreve o roteiro e a estória do jogo. O roteiro é a seqüência de eventos que o jogo executa e a estória cria o contexto no qual o jogo está inserido, além de fazer a ligação entre todos os elementos do jogo: personagens, ambientes e eventos.

A necessidade de equipes específicas de criação de jogos depende da complexidade e do gênero do jogo. Jogos cinematográficos, tais como Resident Evil 4 (GC/PS2) e Doom3 (PC/Xbox), tendem a exigir um grupo de pessoas específicas para criar roteiros e a estória (*script designers*). Jogos que possuem grandes cenários com dezenas (e até centenas) de elementos – inimigos, obstáculos, itens, armadilhas –, tais como God of War (PS2) e Ninja Gaiden (Xbobx), tendem a exigir uma equipe específica de *level design*. A configuração típica de uma equipe de jogos atual – plataformas PS2, GC e Xbox – é mostrada na Tabela 1.

3.3.1 A evolução dos conceitos do jogo: o Ciclo Iterativo de Criação de jogos

No desenvolvimento de jogos, a criação de jogos possui um processo distinto. Assim como todo processo de criação, ele é fortemente iterativo e incremental.

Segundo [PEDERSEN 2003], o processo de criação de um jogo deve envolver jogadores o quanto antes. A cada etapa de desenvolvimento de seus conceitos, o jogo deve ser testado e os resultados são usados para realizar melhorias. Assim, o processo força uma evolução iterativa e incremental dos conceitos do jogo.

Protótipos físicos podem ser construídos para testar os conceitos do jogo em evolução. FULLERTON (2004) sugere a preparação de protótipos antes do início da implementação do software. Protótipos físicos geralmente têm como matéria prima caneta, papel, cartões e até pessoas desempenhando papéis de personagens (ver Figura 4).

Figura 4: Produção e testes de protótipo [FULLERTON 2004].

Através da construção de protótipos físicos e, posteriormente, de protótipos de software, é possível diminuir os riscos de criar um *gameplay* falho. Por meio de uma abordagem iterativa contínua de construção, testes, avaliação e melhoramentos, o processo de criação de jogos evolui o conceito do jogo à sua melhor forma (Diagrama 2). Essa abordagem é aplicada ao longo de todas as etapas de concepção do jogo. O fluxo abaixo demonstra o chamado “ciclo iterativo de criação de jogos”:

1. O conceito é criado.

2. O conceito é formalizado (ex.: documentado ou criado protótipo).
3. O conceito é testado.
4. Os resultados são avaliados, categorizados e priorizados.
5. Se os resultados são negativos e o conceito é fundamental para o jogo, alterações são feitas e o ciclo é reiniciado sobre o mesmo conceito.
6. Se os resultados mostram melhorias, estas são realizadas e ocorre o re-teste.
7. Se os resultados são positivos e o conceito aprovado com sucesso, o processo iterativo sobre aquele conceito está terminado.

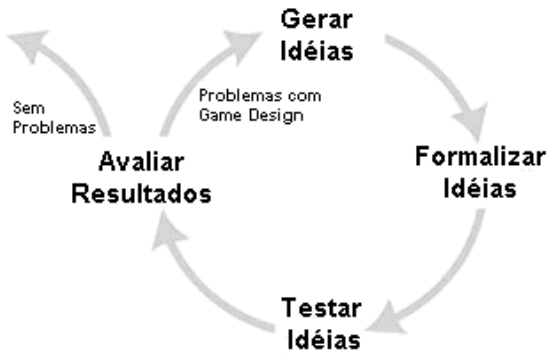

Diagrama 2: O ciclo iterativo de Criação de jogos [FULLERTON 2004].

3.3.2 O Processo de Criação de jogos

O processo de criação de jogos proposto por [FULLERTON 2004] possui sete etapas (abaixo). Ao longo destas, é executado o ciclo iterativo de criação - geração, formalização, testes e avaliação dos conceitos:

Etapa 1: ***"Brainstorming"***: Nesta etapa, todas as idéias são concebidas, desconsiderando detalhes específicos. Todas elas são anotadas e ordenadas em termos de sua relevância. Segundo [FULLERTON 2004], as mais importantes (geralmente, sumarizam-se em três) são descritas em mais detalhes com um pequeno texto (ex.: um parágrafo).

Etapa 2: **Protótipo Físico**: Um protótipo físico é criado usando materiais como papel, caneta e elementos que possam ajudar a representar as regras do jogo (dados, bonecos,

etc.). Este protótipo é testado usando-se o método iterativo (Diagrama 2). Após a validação do protótipo, esta primeira concepção do jogo é documentada em um pequeno texto (ex.: três a seis páginas), descrevendo o funcionamento da mecânica central do jogo.

Etapa 3: **Apresentação**: Nesta etapa é preparada uma apresentação da idéia (Documento de Criação de jogos Conceitual) – com imagens conceituais e descrição dos principais elementos do *gameplay* –, que é feita para o time, a gerência e o financiador do projeto, a fim de garantir os recursos necessários para construção de um protótipo em software. Embora esta etapa seja considerada opcional, ajuda a melhorar a definição do jogo. Neste momento, existe uma validação do jogo pelo financiador. Caso o não haja aprovado, retorna-se ao primeiro passo, realizando-se alterações ou recomeçando com uma nova idéia.

Etapa 4: **Protótipo de Software**: Montando o time principal de desenvolvimento, um protótipo de software é criado. Esse protótipo modela o *core gameplay*, contendo imagens temporárias, que muitas vezes são reutilizadas de outros jogos ou simples rascunhos. Desta forma, é produzida uma primeira versão jogável a baixo custo e tempo. O jogo é testado usando o ciclo iterativo (seção 3.3.1) e depois de aperfeiçoado, é documento (Especificação do jogo).

O protótipo de software pode ser produzido de duas formas: utilizando a tecnologia-alvo do jogo final ou por meio de soluções de construção rápida de protótipos. Dado objetivo deste protótipo – validar o conceito do *core gameplay* – é preferível usar ferramentas que permitam produzir rapidamente resultados. Exemplos de ferramentas de produção rápida são o Macromedia Flash [FLASH 2005], o Game Maker [GAME MAKER 2005] (para jogos 2D) e a Virtools [VIRTOOLS 2005] (para jogos 3D).

Etapa 5: **Documento de Design**: Durante as etapas anteriores, o Criador de Jogos faz

anotações para a criação do jogo final. Essas informações, somadas ao conceito inicial já validade do jogo, são usadas a concepção da primeira versão do documento de Especificação do Jogo.

Etapa 6: **Produção**: Em um primeiro momento, o Criador de Jogos trabalho junto a outros integrantes da equipe para realizar estudos de viabilidade de cada elemento do jogo. Além disso, nesse momento deve-se garantir que elemento esteja corretamente descrito no documento. O objetivo é aprimorar a Especificação do Jogo até que esteja apta a servir como guia para as atividades de implementação de software, arte e mídia..

Com a primeira versão completa da Especificação do Jogo e, a produção do jogo é iniciada. O ciclo iterativo de testes, validações e modificações de criação de jogos também é executado ao longo desta etapa.

Teoricamente os maiores problemas no conceito do jogo devem ter sido resolvidos durante a validação dos protótipos. Desta forma, o número de problemas encontrados e mudanças necessárias devem se tornar menores a cada iteração do projeto. Durante a produção do jogo elementos primordiais modificados não devem ser modificados: esta é uma etapa de pequenas alterações e melhoramentos.

Etapa 7: **Qualidade**: A etapa final do trabalho de um Criador de Jogos está nos testes de qualidade. Neste momento, o jogo deve estar com um *gameplay* completo. Caso alguns pequenos problemas ainda apareçam (é comum que ocorram), o mesmo ciclo iterativo deve ser usado. Devem-se verificar detalhes de usabilidade (incluindo manuais e tutoriais), pois neste momento, o jogo precisa estar acessível ao seu público-alvo.

O ciclo iterativo de criação de jogos é usado em todas as etapas do desenvolvimento de um jogo. Por meio deste processo, todos os elementos do jogo são testados e aprimorados. Este é um processo de criação de jogos genérico, podendo ser aplicado aos tipos de jogos

existentes. Obviamente, nem todas as etapas são realizadas em todos os projetos, pois fatores como complexidade do jogo, tamanho de equipe e prazos podem exigir a exclusão de uma ou duas etapas de apoio para concentrar esforços em etapas de produção.

3.4 Processo de Desenvolvimento de Jogos

O processo de desenvolvimento de jogos segue o mesmo princípio dos processos adotados na produção de outros tipos de softwares: uma seqüência lógica de etapas, partindo da concepção, realizando um planejamento detalhado, passando à produção e chegando à finalização. Neste processo, existem iterações, que executadas consecutivamente baseiam-se em retornos, levando a uma evolução do software (construção incremental).

3.4.1 Papéis, Atividades e Artefatos

Pode-se definir o termo Desenvolvimento de Jogos como o conjunto formado pelas várias áreas da ciência, arte e tecnologia que têm aplicação ou são especificamente destinadas à construção de jogos eletrônicos. Profissionais de diferentes áreas podem fazer parte de um time de produção de jogos, tais como, artistas gráficos, sonoplastas e criadores de jogos.

Jogos de simulações de vôo, corrida ou combate são exemplos de projetos que necessitam de profissionais de áreas distintas executando papéis dentro da equipe. Exemplos comuns são consultores militares e automobilísticos, que trabalharam junto à equipe de criação para adicionar mais realismo ao jogo, e diretores de cena, atores e dubladores de personagens, que trabalham na produção de vídeos, cenas da estória, falas e movimentos dos personagens no jogo. Esses papéis são adicionais e específicos do domínio do jogo sendo produzido, e muitas vezes, esses profissionais são alocados separadamente do grupo principal de produção do jogo.

Na Figura 5 é apresentada uma descrição genérica e subdividida dos vários papéis que podem fazer parte de uma equipe de produção de jogos. Obviamente, essa configuração pode mudar de acordo as características de um projeto. Por serem advindos de áreas diferentes,

determinados conjuntos de papéis possuem um processo de atuação bastante peculiar.

Uma equipe de produção completa de jogos pode ser considerada como a união da equipe de desenvolvimento (pertencente à empresa que desenvolve o jogo) e da equipe de publicação (pertencente à empresa publicadora). A interface entre as equipes se dá, geralmente, por meio dos produtores de ambas as equipes. Um esquema dessa união é mostrado na Figura 5.

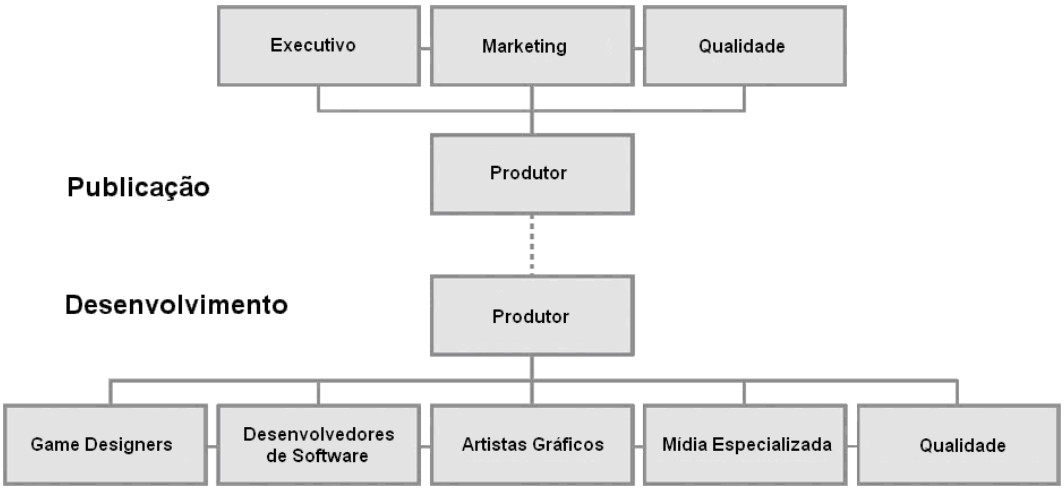

Figura 5: O time de produção de um jogo [FULLERTON 2004].

Os papéis envolvidos no processo de publicação de um jogo estão fora do escopo dessa discussão. Portanto, ao longo desse documento o termo produção é também usado para se referenciar ao pessoal que desenvolve o jogo. Na Figura 5 é possível observar os papéis típicos que estão envolvidos no desenvolvimento de um jogo. Esse modelo de papéis apresenta uma configuração genérica das especialidades de profissionais envolvidos na produção de um jogo.

3.4.1.1 Produtor

No desenvolvimento de jogos, o produtor representa o mesmo papel de um Gerente de Projetos. Este papel não sofre alterações quanto ao modelo do RUP, apenas estando inserido em um domínio diferente. O produtor é responsável pelo planejamento e monitoramento da

progressão do projeto, além de interagir com pessoas externas ao time de desenvolvimento, como clientes e profissionais terceirizados.

O produtor garante o fluxo de informação a todos os integrantes do projeto e também assegura que todo o ambiente de trabalho necessário esteja disponível. O principal artefato pelo qual o Produtor é responsável é o Plano do Projeto.

3.4.1.2 Criadores de Jogos

Os Criadores de Jogos são responsáveis por definir toda a especificação do jogo. Em analogia aos sistemas de informação, um Criador de Jogos assume o papel de um analista de negócios. É responsável por definir em detalhes todos os elementos de um jogo, tais como regras, objetivos, personagens, mecânicas, telas, ambientes, sons, imagens e interações.

Os Criadores de Jogos possuem um papel chave no processo de desenvolvimento de jogos. Dependendo da complexidade, o papel de Criador de Jogos pode ser exercido por um grupo. Dentro deste grupo, papéis mais específicos podem ser melhor distribuídos, tais como: Criador de Níveis, Criador de Personagens e Criador de Estória e Roteiro. Então, um Líder de Criação de Jogos é definido.

O Criador de Jogos é o responsável pela geração do documento de maior importância para o processo de jogos: a Especificação do Jogo (também conhecido como *Game Design Bible* ou *Game Design Especification*, [PEDERSEN 2003]). Esse artefato contem uma descrição amplamente detalhada do jogo. Outro documento de relevância para o papel do Criador de Jogos é o plano de testes de jogabilidade, executado pelos integrantes do time de Qualidade de projeto.

3.4.1.3 Desenvolvedores de Software

O desenvolvimento de software inclui papéis típicos, tais como, analistas, projetistas e programadores. Em projetos pequenos, vários papéis podem ser assumidos por um desenvolvedor de software. Já em projetos maiores, é necessário que pessoas mais experientes

ocupem posições de planejamento e elaboração de soluções (analistas e projetistas) a fim de garantir que o grupo de programadores esteja em sintonia com o plano de desenvolvimento e focado na implementação de complexos algoritmos.

Os desenvolvedores são responsáveis por todo o software do jogo, incluindo suas especificações, arquitetura, planejamento, implementações e testes unitários. Assim como o Produtor, os papéis de desenvolvedores de software são semelhantes ao modelo apresentado no RUP. As diferenças estão no tipo de conhecimento necessário, relacionado ao domínio de jogos.

3.4.1.4 Artistas gráficos

Juntamente com a Criação de Jogos, a Arte Gráfica e as Mídias Especializadas representam a grande diferença do desenvolvimento de jogos em relação a outros softwares.

Os artistas gráficos são responsáveis por toda a parte visual do jogo, que inclui desenhos conceituais, ilustrações de personagens, objetos, cenários e efeitos especiais. Animações também fazem parte das atribuições dos artistas. Quando o projeto é complexo, um grupo e um líder de arte são definidos.

Atualmente, as atividades de arte gráfica demandam maior esforço dentro de um projeto de jogos, representando a parte mais numerosa de profissionais dentro da equipe. Exemplos de papéis de arte comumente encontrados em projetos atuais são desenhistas 2D, modeladores e animadores 3D e artistas conceituais, que trabalham com os conceitos iniciais de cada elemento do jogo. A arte conceitual é geralmente incluída na Especificação do Jogo.

3.4.1.5 Mídia Especializada

O grupo descrito como mídia especializada é geralmente composto por sonoplastas, consultores e dubladores (para as vozes de personagens). Os sonoplastas são os profissionais de mídia especializada necessários a todo projeto de jogos.

O papel de um sonoplasta é produzir os efeitos sonoros e músicas de um jogo.

Geralmente, seu trabalho inicia após alguma parte do jogo estar desenvolvida. Isso ocorre porque o som deve contemplar e enriquecer o ambiente existente do jogo (ajudando na imersão do jogador).

Outros tipos de papéis de mídia especializada comuns em projetos de jogos são consultores. Em projetos de jogos complexos, consultores são contratados para ajudar a tornar o ambiente do jogo mais realista. Consultores são comumente usados em grandes produções de jogos dos gêneros guerra, corrida, luta e dança.

Dubladores são contratados para adicionar falas aos jogos, muito comuns nos jogos atuais. Também são comuns os vídeos de atores reais e animações em desenho ou computadorizadas, que ajudam a contar a estória do jogo. Nestes casos, atores podem ser contratados para atuar como personagens na captura de movimentos, assim como também, os técnicos necessários a operar corretamente os dispositivos de captura.

Músicos, técnicos de efeitos sonoros, consultores, dubladores, atores e técnicos especializados são alguns dos tipos de papéis de mídia especializada comumente integrados as equipes de produção de jogos.

3.4.1.6 Qualidade

O time de qualidade de um projeto de jogos é responsável por todo o projeto, com enfoque no jogo. As principais atividades exercidas por seus integrantes estão relacionadas a testes. São realizados testes estruturais e testes unitários, dentre outros. Papéis como os de Planejador de Testes e Executor de Testes podem ser assumidos por diferentes integrantes da equipe de qualidade ou de produção.

No desenvolvimento de jogos existe uma forma especial de testes: o Teste de Jogabilidade (*playability test*) ([ROUSE 2000]). O objetivo é verificar a consistência do *gameplay* sobre a Especificação do Jogo, fornecendo retorno valioso a todos os outros integrantes da equipe de produção, e especialmente, ao Criador de Jogos. Um dos critérios

fundamentalmente verificados em um teste de jogabilidade é a diversão. Comumente, informações de testes de jogabilidade são obtidas por meio de questionários e/ou observações.

Os testes de jogabilidade são geralmente executados por pessoas selecionadas com base em seu "perfil de jogador". Isso ocorre porque os jogos são desenvolvidos considerando-se determinadas características do público-alvo, tais como faixa etária, sexo, e perfil de jogador – assíduo, freqüente ou casual. Estes testes possibilitam aos desenvolvedores obter críticas, sugestões e opiniões de jogadores reais.

A classificação dos jogadores em três perfis segue critérios como interesse, freqüência de uso e disponibilidade a investimentos. O jogador assíduo, popularmente denominado gamemaníaco, é aquele que tem nos jogos sua principal forma de entretenimento. Geralmente grande parte de seu tempo é usada tanto. Ele está disposto a investir o tempo e dinheiro que se fizerem necessários para garantir seu entretenimento.

Popularmente chamado de "gamer", o jogador freqüente, tal qual o assíduo, possui grande interesse em videogames. No entanto, é mais criterioso na escolha de jogos e acessórios. Jogos são sua forma de entretenimento favorita, mas destina uma menor parte de seu tempo para os videogames quando comparado ao perfil assíduo.

Jogadores assíduos e freqüentes fazem mais uso de jogos complexos, considerados o estado da arte. Também preferem jogos intelectualmente e habilidosamente desafiadores.

Por fim, o jogador casual é aquele que se utiliza de videogames de forma esporádica. É o maior usuário de jogos simples: promocionais, educativos ou contidos na Internet. Em oposição aos perfis assíduo e freqüente, não tem interesse em jogos complexos e desafiadores, que afastam seu interesse pela necessidade de minutos ou até horas para serem compreendidos e utilizados de forma correta.

3.4.2 Etapas de Produção de um Jogo

O desenvolvimento de jogos é realizado através de etapas bem definidas: parte de uma concepção inicial da idéia, passa por um detalhado planejamento e preparação, segue com a construção do software, arte, mídias (como músicas e sons) e níveis, e tem seu término na pós-produção, onde pequenos ajustes e correções são realizados. Essas etapas são mostradas no Diagrama 3 e os papéis, atividades e artefatos envolvidos em cada uma são discutidos a seguir.

O processo descrito pode ser considerado genérico, demonstrando a seqüência lógica de passos necessários à produção de um jogo e desconsiderando a iteratividade. No entanto, o desenvolvimento de jogos e o processo de criação de jogos são fortemente iterativos e incrementais. A descrição de suas etapas é feita desta forma com o intuito de facilitar a compressão geral do processo.

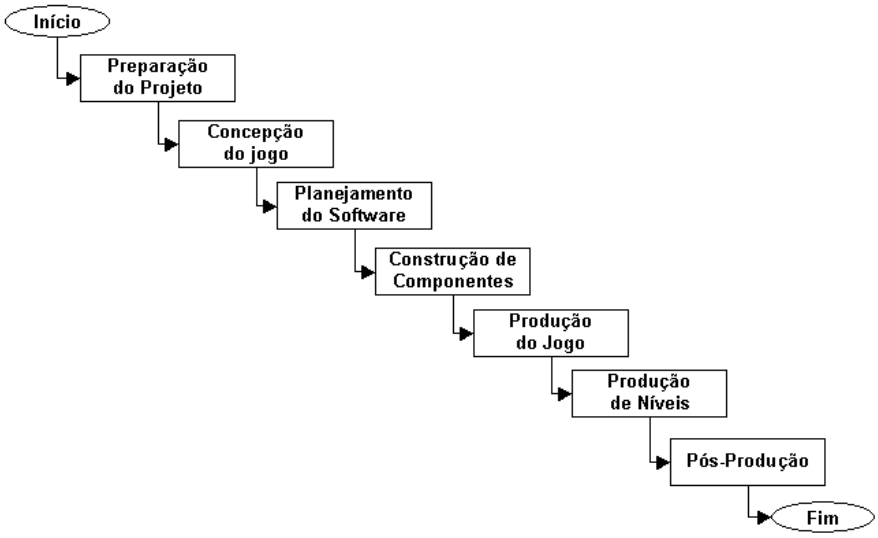

Diagrama 3: Etapas de produção de um jogo [FLOOD 2003].

1. Preparação do Projeto

- Criação do Conceito Inicial (Especificação conceitual do jogo, com foco em marketing e diferenciais da idéia).
- Compreende as três primeiras etapas do processo de criação de jogos

(*Brainstorming*, Protótipo Físico e Apresentação - Seção 3.3.2).

- Preparação do Contrato e validação do cliente para seguir com o projeto.
- Estudos iniciais de viabilidade.

2. Concepção do Jogo

- Planejamento final do projeto.
- Planejamento do Jogo em termos de Criação de jogos, Arte e Mídia (inclui planejamento de marcos do projeto).
- Verificação de Viabilidade mais completa e detalhada.
- Validação dos conceitos com criação de protótipo em software.
- Compreende as etapas de Protótipo de Software e Especificação do jogo do processo de Criação de jogos (Seção 3.3.2).

3. Planejamento do Software

- Avaliação das soluções necessárias para os recursos do jogo baseando-se na Especificação do jogo.
- Captura de Requisitos. Esta atividade (e seu artefato) é considerada opcional para alguns jogos. Projetos menores podem usar a Especificação do jogo.
- Planejamento do Software (arquitetura, soluções e marcos).

4. Construção de Componentes

- Construção da arquitetura básica do jogo:
  - *Frameworks* de software.
  - *Engine* do Jogo.
  - Ferramentas.

- Os elementos desta arquitetura básica podem ser construídos ou reutilizados (e adaptados) de outros projetos (ou de terceiros).

5. Produção do Jogo

- Produção do jogo (software, arte e mídia) de acordo com o planejamento anterior.
- Testes funcionais nas rotinas e módulos do software.
- Testes de jogabilidade realizados pelo próprio pessoal da equipe. Estes são testes minuciosos sobre o *gameplay* e podem ser feitas mudanças consideráveis na mecânica do jogo.
- A produção é dividida em vários lançamentos (mini-projetos) internos e externos.
- A Especificação do jogo é constantemente verificada e atualizada a medida que surgem novas idéias.
- Compreende a etapa de Produção do processo de Criação de jogos (Seção 3.3.2).
- Ferramentas construídas na etapa anterior são atualizadas de acordo com necessidades novas.
- A produção de mídia é iniciada: músicas e efeitos.

6. Produção de Níveis

- Todos os cenários do jogo são construídos.
- As pendências necessárias para a realização desta atividade são resolvidas: elementos do jogo, partes do software ou ferramentas.
- Especificação do jogo é sempre verificada de atualizada de acordo com novas idéias e retornos da própria equipe de desenvolvimento.
- Também compreende a etapa de Produção do processo de Criação de jogos (Seção 3.3.2)..

- O restante da parte de mídia do jogo é produzido: músicas, efeitos, vozes e vídeos.
- São produzidos os manuais e material promocional do jogo.

7. Pós-Produção

- Compreende a etapa de Qualidade do processo de Criação de jogos (Seção 3.3.2).
- São realizados testes, validações, pequenos ajustes, modificações e correções de problemas.
- Testes *Alpha*: Realizados com pessoal selecionado – avaliadores especialistas. Mudanças simples são possíveis
- Testes *Beta*: Realizados com usuários finais – jogadores que não possuem conhecimento prévio do jogo. Podem acusar pequenas falhas de usabilidade de interface (diagramação de menus, indicadores do jogo e telas).
- Muito embora a etapa de Pós-Produção seja fortemente baseada em testes, estes são realizados ao longo de todo o ciclo pelo Criador de Jogos para validações e melhorias no conceito (Ciclo Iterativo de Criação de jogos – Seção 3.3.1).

3.5 Tecnologias Relacionadas

As plataformas de desenvolvimento de aplicativos móveis atuais demonstram as peculiaridades deste modelo de produção. Existem cinco plataformas:

- J2ME [J2ME 2005]
- Brew [BREW 2005]
- Mophun [MOPHUN 2005]
- In-Fusio [IN-FUSIO 2005]
- Flash Lite [FLASH LITE 2005]

Para a execução de aplicações nestas plataformas, é preciso que o telefone celular

possua o *midleware* instalado pelo fabricante do aparelho.

Um subconjunto das plataformas utiliza a linguagem Java (J2ME e In-Fusio) e o outro, a linguagem C/C++ (Brew e Mophun). A *Flash Lite* utiliza um script próprio (Action Script) que herda a sintaxe da linguagem C. As que utilizam a linguagem C/C++ possuem compiladores que geram código nativo para o processador central do dispositivo. A J2ME utiliza uma máquina virtual reduzida chamada de KVM (*Kilobyte Virtual Machine*), com menos recursos quando comparada às JVM's (*Java Virtual Machines*) de *desktops*. Embora exista uma pequena máquina virtual rodando por baixo de um *Midlet Java* (aplicativo J2ME), é difícil mensurar a possível perda de desempenho por ser uma tecnologia interpretada.

Desenvolvida pela *Sun Microsysems*, a plataforma J2ME é a mais popular, disponível na maior parte dos celulares que executam aplicações. Ela utiliza a linguagem Java e uma API específica para construção de Midlets (aplicativos móveis).

A segunda plataforma mais usada é Brew – *Binary Runtime Environment for Wireless* –, mantida pela Qualcomm, proprietária da tecnologia CDMA. A plataforma define também um modelo de negócios específico. Brew utiliza a linguagem C/C++ e um compilador ARM, gerando código nativo para o processador central do dispositivo. O modelo de comercialização centralizado e a promessa de melhor desempenho da compilação são as duas maiores vantagens da plataforma.

A plataforma *Flash Lite* é mais recente. Ela é uma versão mais simplificada do Flash, usando uma máquina virtual chamada *Flash Player Lite*. Possui sofisticados recursos para criação de aplicativos multimídia, como jogos e animações, além de extensões específicas para o desenvolvimento móvel, permitindo acessar recursos como envio de mensagens SMS, execução de chamadas, vibração e iluminação da tela.

A plataforma *Mophun* possui menor suporte em número de aparelhos celulares. Ela usa C/C++ com código compilado. A RTE (*Run Time Engine*) é a máquina de execução das aplicações Mophun. Ela precisa estar pré-instalada no dispositivo. Apenas alguns aparelhos

antigos das fabricantes Ericsson e Nokia possuem a RTE.

3.6 Sumário

Nas primeiras gerações de jogos computadorizados, os processos de desenvolvimento e o de criação de jogos eram tidos como “estado da arte”. Com o crescimento de esforços e investimentos necessários para a produção de um jogo, o Processo Unificado da Rational passou a ser o principal processo usado como base nessa indústria.

O processo de desenvolvimento de jogos pode ser visto de uma forma seqüencial, embora existam várias iterações para o refinamento do *gameplay*. Estas são causadas pelo ciclo iterativo de criação de jogos, crucial para se alcançar um jogo de qualidade. Como área multidisciplinar, o desenvolvimento de jogos precisa de um processo que permita flexibilidade – para uma tomada rápida de decisões – e controle – sobre o ciclo de produção. Essas necessidades são discutidas no próximo capítulo.

4 Processo Unificado da Rational e Jogos Móveis

4.1 Introdução

O RUP é um *framework* de processo bastante extenso. Desta forma, é necessário apontar um cenário inicial que possa ser usado como base para aplicação em jogos móveis.

O objetivo deste capítulo é apresentar características do desenvolvimento de jogos móveis que serão usadas para definir um cenário de processo a ser tomado por base no próximo capítulo, onde é discutida a adequação do RUP para jogos móveis. Práticas de *Extreme programming* (XP) são discutidas quanto ao seu uso em RUP.

4.2 Características de Processo para Jogos Móveis

Na indústria de jogos, existe uma constante preocupação sobre como a complexidade dos projetos afeta o processo de produção. Essa preocupação está principalmente sobre gerenciamento de requisitos, tamanho das equipes e tempo de produção, fatores que vêm aumentando a cada nova geração[1] de consoles de videogames lançados no mercado.

Segundo [FULLERTON 2004] e [ROLLINGS 2003], à medida que o hardware dos videogames e computadores evolui, a complexidade dos jogos cresce de forma proporcional. Conseqüentemente, há um aumento no esforço, tempo de produção, número de requisitos e quantidade de recursos e pessoas necessários para a construção do jogo.

Baseando-se em estatísticas de projetos de jogos em videogames, [FULLERTON 2004] apresenta um histórico das média de “tamanho de equipe” e “tempo de produção” por geração de console (Gráfico 1, Gráfico 2 e Tabela 1).

[1] Uma geração de consoles de videogames classifica uma família de dispositivos com recursos semelhantes.

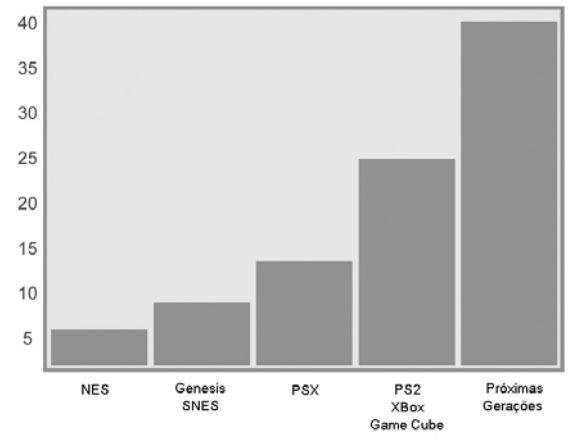

Gráfico 1: Tamanho médio de equipes por plataforma [FULLERTON 2004].

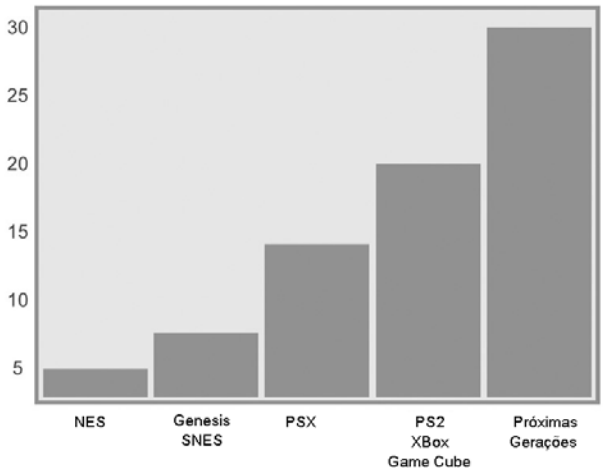

Gráfico 2: Tempo médio de produção por plataforma (em meses) [FULLERTON 2004].

No mercado de videogames portáteis e jogos móveis, existe um atraso de uma geração em relação aos consoles domésticos e computadores pessoais. Isso ocorre porque os dispositivos possuem menor quantidade de recursos computacionais..

Atualmente, existem duas plataformas que dividem o mercado de desenvolvimento de jogos para celular: J2ME e Brew. A plataforma J2ME (*Java 2 Micro Edition*) da Sun Microsystems é a mais usada para jogos 2D, que em geral possuem menor complexidade, quando comparados à jogos 3D. A plataforma Brew (*Binary Runtime Environment for Wireless*) da Qualcomm têm sido bastante usada em jogos 3D, principalmente por grandes empresas.

Comparando-se a complexidade dos jogos de celular atuais com os jogos para

consoles de videogames pode-se dizer que, em sua maioria, os jogos móveis 2D são semelhantes ou inferiores em complexidade a um jogo típico da geração do console Nintendo NES (*Nintendo Entertainment System*). Embora o hardware disponível em celulares seja bastante superior ao de um NES (processador de 8 bits) os jogos móveis atuais são produzidos com foco em jogadores casuais. Uma parte menor dos jogos móveis são mais sofisticados, com a mesma complexidade de desenvolvimento de jogos da geração dos consoles Sega Genesis e Nintendo SNES (*Super Nintendo*) (processadores de 16 bits). É o caso dos jogos móveis produzidos por empresas como Gameloft, Jamdat, Sega e Electronic Arts.

Tabela 1: Configuração média de papéis por plataforma [FULLERTON 2004].

Plataforma	Configuração da Equipe
NES	• 1 Produtor/Game Designer • 2 Programadores • 3 Artistas
Genesis/SNES	• 1 Game Designer • 1 Produtor • 3 Programadores • 4 Artistas

Playstation (PSX)	• 1 Líder de Game Design
	  o 2 Level Designers
	• 1 Produtor
	  o 1 Assistente de Produção
	• 1 Líder de Programação
	  o 3 Programadores
	• 1 Líder de Arte
	  o 4 Artistas

PS2/GameCube/Xbox	• 1 Líder de Game Design
	  o 4 Level Designers
	• 1 Produtor
	  o 1 Assistente de Produção
	• 1 Líder de Programação
	  o 2 Programadores de Engine
	  o 4 Programadores de Jogo
	• 1 Líder de Arte
	  o 10 Artistas

Próxima Geração	• 1 Diretor de Game Design
	  o 2 Game Designers
	  o 4 Level Designers
	• 1 Produtor Executivo
	  o 2 Produtores
	  o 1 Assistente de Produção
	• 1 Líder de Programação (Engine)
	  o 3 Programadores
	• 1 Líder de Programação (Jogo)
	  o 6 Programadores
	• 1 Diretor de Arte
	  o 3 Líderes de Arte
	  o 14 Artistas

Os jogos 3D estão um passo à frente em relação à complexidade, quando comparados aos 2D. Pode-se dizer que em média, os jogos móveis 3D têm complexidade abaixo de um Sony PSX (*Playstation*), mas algumas empresas como Capcom, Electronica Arts e Gameloft e Jamdat já lançaram versões de jogos quase idênticas às de PSX para celulares.

Atualmente, os jogos móveis 2D ainda são maioria, visto que aparelhos que têm suporte nativo à instruções 3D ainda apresentam um custo alto para se popularizarem. Obviamente, em um futuro breve, o mercado de jogos móveis seguirá a mesma evolução dos consoles de videogames, e jogos 3D se tornarão cada vez mais populares, embora muitos ainda apostem no paradigma 2D para jogadores casuais, que representam a maior parte do mercado de jogos móveis.

De acordo com observações feitas nos trabalhos de [RUCKER 2002], [LAM 2004], [WELLS 2004], [BARBAGALLO 2003] e [FULLERTON 2004], pode-se inferir um conjunto de características comuns à projetos de desenvolvimento de jogos e, especificamente, de jogos móveis:

- Como todo projeto de jogos, é comum que mudanças ocorram ao longo do projeto, geralmente causadas por alterações no conceito do jogo ou em sua tecnologia.
- Diferente dos jogos para plataformas atuais mais sofisticadas, como PC e console de videogame, os prazos de jogos móveis são geralmente menores. Pode-se dizer que variam entre 3 a 10 meses, em contrapartida aos projetos de 20 a 30 meses de consoles de última geração (Gráfico 2).
- Os conceitos dos jogos são geralmente mais simples que os de plataformas mais sofisticadas. Em conseqüência, pode-se dizer que as equipes variam em tamanho de pequenas a médias (Gráfico 1). Também, os custos e investimentos são menores.
- A interatividade ocorre em todos os níveis da equipe de desenvolvimento, uma vez que qualidade em jogos não significa software funcionando, sem defeitos e que cumpra todos os requisitos: qualidade em jogos significa ter um jogo atraente e

divertido. Isso torna a qualidade em jogos algo mais abstrato e por esta razão, o processo de criação de jogos exige comunicação constante entre todos os membros da equipe para que se tenha uma visão única do conceito sendo desenvolvido.

De acordo com o processo de desenvolvimento de jogos apresentado em "Desenvolvimento de Jogos" (Capítulo 3), e também segundo as características apresentadas acima, pode-se inferir peculiaridades de um processo de jogos móveis:

- Existe menor produção de artefatos que não sejam o próprio software (jogo).
- Mudanças no escopo (conceito do jogo) ocorrem ao longo do ciclo de desenvolvimento.
- Existe necessidade de agilidade e flexibilidade para que mudanças na concepção do jogo possam ser rapidamente implementadas sem causar atrasos ou problemas.
- O processo de criação de jogos é intuitivamente iterativo e fortemente apoiado em retornos. Logo, espera-se um processo que permita iteratividade e adaptações.

4.2.1 Necessidades de um Processo de Jogos

A produção de jogos exige certas características de um processo que são consideradas na definição do processo de jogos móveis proposto. As principais delas são: agilidade, desenvolvimento iterativo incremental (evolutivo), utilização de documentação (com certo grau de formalidade), preocupação em antecipar uma arquitetura executável e desenvolvimento guiado pela Especificação de Jogos.

Os processos de criação e implementação de um jogo são naturalmente evolutivos. Um bom *gameplay* só é atingido depois de vários testes sobre o jogo em desenvolvimento. Os retornos desses testes (formais ou informais) são usados para modificar o jogo e evoluir a parte fundamental do *core gameplay*. Adicionalmente, o próprio processo de criação de jogos é iterativo e incremental. Desta forma, fica claro que um processo de produção de jogos precisa permitir mudanças (controladas) ao longo do ciclo de desenvolvimento. Também,

precisa utilizar uma abordagem iterativa incremental a fim de permitir que o *gameplay* (e o próprio jogo em implementação) possa evoluir até alcançar o conceito de “divertido”.

Vale ressaltar que, a produção de um jogo geralmente envolve duas empresas: a Desenvolvedora, responsável por criar e implementar o jogo e; a Publicadora, responsável por financiar o projeto e, distribuir e vender o jogo ao consumidor final. É comum que a Publicadora exija documentação formal da Desenvolvedora, seja para fins comerciais ou para avaliação da competência da empresa. Também, a Publicadora exige uma forma de rastreamento da progressão do trabalho da Desenvolvedora: é preciso um ciclo de desenvolvimento que demonstre claramente qual etapa está sendo executada naquele momento, quais são as produções sendo realizadas e quais etapas já foram cumpridas e as outras que estão por vir.

Um elemento fundamental do software de um jogo é a arquitetura. Pode-se descrever a implementação de um jogo como um processo dividido em três etapas: desenvolvimento de uma arquitetura executável, desenvolvimento do *core gameplay* e implementação de todos os elementos do jogo.

O termo arquitetura executável em um jogo corresponde ao “motor” do software: a chamada *engine*. Os jogos atuais são todos desenvolvidos sobre uma *engine*. Ele é responsável por disponibilizar e gerenciar todas as funcionalidades primárias do software: aplicação, memória, acesso a arquivos, entradas do usuário, projeção gráfica, som, inteligência artificial, física simuladas e etc. Sem a *engine*, o trabalho de implementação do *core gameplay* e do restante do jogo se torna muito custoso. Além disso, uma *engine* usualmente provê ferramentas de produção necessárias ao desenvolvedor do jogo: editor de níveis (cenários), visualizador de objetos ou personagens, ambientes de programação e depuração, assim como, linguagens de script para desenvolvimento da lógica, da física e da inteligência artificial do jogo. Um segmento das empresas tornou-se especialista em desenvolver *engines*, enquanto outras os utilizam para produzir os jogos.

A *engine* também define a arquitetura da aplicação do jogo. Desta forma, pode-se dizer que o processo de um jogo é centrado no desenvolvimento de arquitetura, e isso precisa ser feito logo no início do projeto. Ao longo do ciclo de desenvolvimento, essa arquitetura evolui para acomodar todos os recursos que o jogo deve ter.

Por fim, um processo de desenvolvimento de jogos pode ser dito como orientado à criação de jogos. Como discutido na seção 3.3, toda produção é baseada nos documentos de especificação do jogo, de responsabilidade da equipe de criação de jogos. Essa documentação guia a definição dos requisitos, o projeto do sistema, o planejamento e realização dos testes e a aprovação do resultado final pela Publicadora.

4.2.2 RUP como um processo para Jogos

A seção anterior destacou as principais necessidades de um processo de desenvolvimento de jogos, que compreende:

- Iterativo e incremental (evolutivo);
- Permitir mudanças;
- Documentação com adequados níveis de formalidade;
- Explicitar a progressão com etapas bem definidas;
- Centrado em arquitetura (*engine* ou *framework*);
- Orientado pela Criação de Jogos.

Segundo ([LARMAN 2003]), as principais abordagens que têm sido aplicadas na produção de software são: RUP, *Extreme Programming* [BECK 2004] - e *Feature-Driven Development* - [LARMAN 2003]. XP e FDD são consideradas metodologias ágeis.

Comparando-se as características de projetos de jogos citadas com as definições de XP e FDD (respectivamente [BECK 2004] e [LARMAN 2003]) é possível afirmar que XP e FDD são metodologias que atendem as necessidades da produção de jogos, embora que de forma não tão completa quanto o RUP. Acredita-se que esta seja uma das principais razões pelas

quais estas metodologias, em especial XP, têm sido empregadas na produção de jogos. No entanto, segundo [FLOOD 2003], [FLYNT 2004] e [RUCKER 2002], o Processo Unificado da Rational é o mais usado nessa área.

De acordo com seus conceitos e definições ([KRUCHTEN 2000], [KROLL & KRUCHTEN 2003], [JACOBSON et. al. 1999] e [LARMAN 2001]), RUP satisfaz com sucesso todos os requisitos de um processo de desenvolvimento de jogos. Uma análise resumida das três metodologias frente a cada aspecto do processo de jogos é apresentada na tabela abaixo.

RUP	XP/FDD	Características de Processos de Jogos
Atende Melhor	Atende	**Iterativo e Incremental** Por definição, RUP é iterativo e incremental (sessão 2.3.1.3) e executado por meio de mini-projetos (iterações). XP e FDD também seguem esta abordagem. No entanto, o modelo de planejamento e controle de execução das iterações do RUP é mais explícito.
Atende	Atende	**Permitir Mudanças** O gerenciamento de mudanças faz parte da essência do RUP (sessão 2.3.2) assim como de XP e FDD.
Atende Melhor	Atende	**Documentação com Níveis de Formalidade Adequados** Pode-se afirmar que o RUP é, seguramente, o processo com maior destaque quando se considerada a documentação. Todos os artefatos do RUP são claramente especificados e possuem diferentes modelos (*templates*) que atendem a diversos níveis de formalidade. XP e FDD, por serem disciplinas ágeis, não possuem foco em documentação. Um correto nível de formalidade aplicado aos artefatos do RUP permite que este se torne um processo bastante ágil, contudo, mantendo o planejamento, controle e flexibilidade necessários a um processo de jogos.
Atende	Não Atende	**Explicitar a progressão com etapas bem definidas** RUP possui etapas de desenvolvimento bem definidas, apresentada por sua a estrutura horizontal, que representa o processo em termos de ciclos, fases iterações e marcos (sessão 2.3.4). A progressão de um

		projeto RUP pode ser facilmente medida através da sucessão de iterações, fases e ciclos. XP e FDD, por outro lado, não definem de forma clara a organização do ciclo de vida do software em termos de etapas.
Atende	Não Atende	**Centrado em Arquitetura** Na produção de jogos, a arquitetura é um elemento-chave de todo o processo. A *engine* é o artefato fundamental para a implementação de um jogo. Tanto XP quanto FDD não explicitam um momento no ciclo de vida no qual a arquitetura precisa ser definida e desenvolvida. Ambas deixam a cargo do desenvolvedor a decisão de quais recursos do software devem ser implementados a cada etapa. RUP possui uma etapa bem definida, que concentra as atividades de planejamento e implementação da arquitetura: a Elaboração. Desde a criação do Documento de Visão no processo, RUP enfatiza que as primeiras características do software que devem ser criadas - investigadas, analisadas, projetadas e produzidas - são aquelas de maior significância para a arquitetura do sistema.
Atende	Atende	**Orientado pela Criação de Jogos** Por este ser um critério somente presente na área de criação de jogos, ele é desconsiderado nesta análise.

Tabela 2: Análise das metodologias frente as características de processos de jogos.

O Processo Unificado da Rational tem sido aplicado na indústria com sucesso há vários anos. É considerada uma metodologia madura que provê suporte - por meio de configurações e especializações - a uma diversa gama de domínios de aplicações. [FULLERTON 2004], [FLYNT 2004], [RUCKER 2002], [BETHKE 2003] e [FLOOD 2003], são alguns dos autores que descrevem abordagens baseadas no RUP usadas com sucesso na indústria de jogos.

As abordagens de *Extreme programming* e *Feature-driven development* não são, em hipótese alguma, descartadas para o emprego na produção de jogos. O objetivo desta investigação é apresentar a proposta de aplicação de um processo de software para o

desenvolvimento de jogos móveis. As características de projetos de jogos apontam RUP como o processo mais adequado para ponto de partida. No entanto, XP e FDD são passíveis de aplicação e constituem um bom tema para estudos futuros (sessão 6.7).

A área de desenvolvimento de jogos (móveis ou não) é ainda muito nova no Brasil. Considera-se que um processo que ofereça formas mais bem definidas e controladas de trabalho seja uma melhor opção quando a maior parte da equipe não tem domínio completo sobre a tecnologia e as técnicas empregadas na produção do software. Neste contexto, RUP leva vantagens, pois de acordo com [LARMAN 2001] e [KRUCHTEN 2000], metodologias ágeis tendem a ser melhores adequadas a equipes mais experientes, que possam desenvolver o trabalho sem precisar de guias ou documentação extensa.

4.2.3 Necessidades de Métodos Ágeis num Processo de Jogos

Embora se tenha o RUP como elemento fundamental desta investigação, considera-se que o uso de elementos de XP adequadamente adaptados pode trazer bons resultados a um processo de jogos baseado em RUP.

Sabe-se que as duas metodologias mais populares são RUP e XP. Ambas têm sido usadas na indústria e discutidas no meio acadêmico, o que pode se pode comprovar pela extensa quantidade de trabalhos e artigos publicados. Elas também já provaram sua validade para cenários específicos. Comumente, RUP é considerado um processo de maior controle, mas também tradicionalista e rígido. XP é descrita como uma metodologia leve, inovadora e ágil, mas possui certas restrições de aplicação e não provê tantos elementos customizáveis quanto RUP.

De acordo com os critérios de processo de jogos levantados e suas análises frente às metodologias, RUP demonstra-se como o mais adequado. XP não define muito bem etapas e artefatos, e em determinadas situações ou organizações, algumas de suas práticas podem tornar-se impraticáveis. Para projetos menores e de curta duração, como jogos 2D simples, a

informalidade de XP é perfeitamente aplicável. No entanto, em projetos mais complexos, a necessidade de intensa comunicação por parte da metodologia pode ser um problema: é comum que nem toda a equipe esteja presente em todo o processo de desenvolvimento. A maior parte do trabalho de criação de jogos está no início no projeto e já o pessoal de implementação, arte e som, começam a trabalhar maciçamente em etapas posteriores.

O RUP abrange uma maior variedade de domínios, pois é considerado um processo base configurável. Obviamente sua forma "natural" seria definitivamente muito "pesada" e rígida para qualquer domínio. A chave para o bom uso de RUP está na definição da configuração adequada para cada projeto. O objetivo desta sessão é obter um cenário inicial de RUP a partir do qual serão adicionados os elementos para aplicação em jogos móveis. Neste sentido, [POLLICE et. al. 2003], [KROLL & KRUCHTEN 2003], [LARMAN 2001], [LARMAN 2003] e [KRUCHTEN 2000] afirmam que é possível utilizar uma abordagem mais ágil de RUP, inclusive implementando práticas de XP.

Como resultado, o cenário considerado para início da discussão sobre jogos é o RUP Lite: uma versão menos "rígida" do processo. A escolha deste RUP Lite como base um processo de jogos advém das constatações documentadas nesta e na sessão anterior: parte-se do princípio de que RUP propõe fases, papéis e artefatos bem definidos e a *Extreme programming* possui práticas produtivas e um modo mais humano de trabalho. Assim, a discussão sobre práticas de XP pode agregar elementos interessantes a uma versão leve de RUP, de forma a contribuir para a produção no domínio de jogos móveis.

4.3 Um cenário de RUP para Jogos Móveis

As características levantadas para a maioria dos projetos de jogos móveis 2D – mudanças constantes, prazos menores e equipes pequenas – ratificam a necessidade de um modelo de RUP leve e ágil. Desta forma, a idéia é produzir uma quantidade mínima de artefatos que não sejam o próprio software, maximizando os resultados perante prazos curtos.

Com o uso dos artefatos corretos consegue-se obter controle sobre o ciclo de desenvolvimento em um nível seguro. Minimizar a produção destes artefatos de controle possibilita que mudanças ocorram e sejam aceitas de forma mais natural, sem impedir a progressão do projeto.

O conceito de um cenário de RUP que implementa elementos XP é aceitável, pois segundo [POLLICE et. al. 2003], [KROLL & KRUCHTEN 2003], [LARMAN 2001], [LARMAN 2003] e [KRUCHTEN 2000], não existem barreiras para o uso de uma abordagem ágil de RUP. As metodologias ágeis possuem vantagens que podem ser melhor observadas durante a etapa de construção de um novo software. A abordagem RUP traz melhores mecanismos de planejamento e controle das atividades, o que pode acrescentar melhorias ao desenvolvimento de jogos móveis. Exemplos destes mecanismos são artefatos, atividades, papéis e diretrizes.

De acordo com [KROLL & KRUCHTEN 2003], o RUP não força o uso de todas as disciplinas de negócio. Ele provê um conjunto de elementos selecionáveis para a realidade de cada projeto ou organização. Desta forma, selecionam-se apenas aqueles que possam tornar o processo mais ágil e iterativo, permitindo assim, a incorporação de práticas de *Extreme programming*.

Os critérios levantados convergem para a definição de um cenário inicial "leve" e ágil do RUP para então se partir para a discussão em torno de sua aplicação para jogos móveis. Esse formato de RUP tem algumas características marcantes:

- Menos rigoroso: prioriza a produção do jogo e diminui a produção de artefatos que não sejam o software. Essa característica está fortemente ligada ao fato de que a qualidade de um jogo está em ele ser atraente e divertido e não somente ao fato de que funciona, passa pelos testes e possui todos os requisitos implementados.
- Corretamente formalizado: nem todos os documentos precisam passar por padrões extremamente formais. No entanto, a formalização é importante para a documentação

e padrozinação do projeto. O que se aponta aqui é o excesso de formalização e de forma alguma sua exclusão. O importante é balancear a formalização com o nível de controle necessário ao projeto. Em projetos curtos e com poucas pessoas, a formalização pode ser baixa. Em jogos 2D mais sofisticados e jogos 3D, a formalização de documentos, em especial para a Especificação do jogo (e requisitos), a documentação técnica e a parte de testes, é bastante importante. Desta forma, é importante que o cenário inicial de RUP permita diferentes níveis de formalização.

- Mais iterativo: a qualidade de um jogo depende muito do conceito do jogo. O processo de criação precisa ser iterativo: ele precisa de experimentos. Encontrar a diversão em um jogo depende de técnica e tentativas. Por esta razão, é importante que o RUP Lite seja essencialmente iterativo. Isto também contribui para a resposta a mudanças ao longo do projeto.

Segundo [KRUCHTEN 2000], o Processo Unificado da Rational deve ser configurado de forma que possa melhor atender às necessidades do projeto. Um processo nunca deve ser o objetivo do trabalho de uma equipe. O engano mais comum na adoção do RUP é adotar muitos de seus artefatos e atividades na configuração definida. Produzir muitos artefatos pode diminuir a produtividade da equipe, pois boa parte do esforço poderá estar sendo despendido na produção de artefatos que podem muitas vezes ser irrelevantes à realidade do projeto. Como conseqüência, consegue-se alcançar uma abordagem mais ágil, que permite a ocorrência de mudanças (baseadas em retorno), elemento considerado chave para todo processo de criação (incluindo o de jogos).

4.3.1 RUP Lite

O cenário inicial considerado para aplicação de RUP em jogos móveis é o RUP Lite. O RUP leve proposto por [POLLICE et. al. 2003], [KROLL & KRUCHTEN 2003], [LARMAN 2001], [LARMAN 2003] e [KRUCHTEN 2000]) é baseado em cenários de

processo reais. Com base nestes trabalhos, pode-se apresentar um RUP sem contradições aos seus conceitos fundamentais. [LARMAN 2003], [POLLICE 2004] e [KROLL 2003] descrevem configurações leves, baseadas na "Essência do RUP" (seção 2.3.2), que descreve os conceitos chave do processo. Essa essência é a base para um cenário leve. Segundo estes trabalhos, toda configuração do RUP deve conter sua essência. Caso contrário, tais processos não podem ser denominados como especializações do RUP.

O objetivo desta seção é apresentar os quatro elementos que caracterizam o cenário de RUP Lite. Eles estão de acordo com as características apresentadas na seção anterior: menos rigoroso, corretamente formalizado e mais iterativo.

- Artefatos-chave do RUP: Os artefatos essenciais estão fortemente ligados a essência do RUP (seção 4.3.1.1). O conceito RUP Lite preza pela diminuição da produção de artefatos que não sejam o próprio software, característica que está presente também na essência do RUP. Esses artefatos formam o conjunto mínimo e essencial para todo processo que use a essência RUP.
- Processo Iterativo e Incremental: Um processo leve é fortemente baseado no modelo iterativo incremental, e conseqüentemente, em retornos. É importante que o RUP Lite não seja similar ao modelo cascata. A iteratividade é importante para a evolução de um projeto e isso é essencial para o conceito de um jogo. Além disso, segundo a definição básica de RUP (seção 2.3.1.3), o processo deve ser iterativo e incremental.
- Níveis de formalidade: Agilidade, foco na produção do software, ênfase na comunicação inter-pessoal, *feeedbacks* e pouca burocracia. Todas essas características estão presentes em modelos de metodologias ágeis e o nível de formalidade na comunicação e na produção de artefatos possui grande influência sobre o desenrolar destas características.
- Práticas ágeis. Segundo autores como [LARMAN 2001], [POLLICE et. al. 2003] e [KROLL & KRUCHTEN 2003], o uso de metodologias ágeis em uma configuração

mais leve do RUP pode agregar bons resultados. Um conjunto de práticas bem conhecidas de XP é discutido para aplicação no RUP Lite (seção 4.3.1.4).

4.3.1.1 Os Artefatos-Chave do RUP

De acordo com [KRUCHTEN 2000], existem artefatos descritos no Processo Unificado da Rational considerados importantes a toda configuração. Esses artefatos estão ligados à essência do RUP, contribuindo para que as atividades de um projeto possam ser realizadas de forma organizada e progressiva.

Uma importante prática do RUP define que os artefatos do projeto devem estar disponíveis a todos os seus participantes. Esta prática é centrada em comunicação: os artefatos são peças-chave no RUP e todos devem ter acesso ao planejamento do projeto, seu progredindo e como será finalizado. Desta forma, há contribuição para que todos possam manter uma visão única e consistente do objetivo do projeto. É considerada uma boa prática disponibilizar um *site* do projeto com todos estes artefatos, permitindo acesso fácil e direto.

Os artefatos considerados chave dentro do RUP estão, em sua maior parte, relacionados ao planejamento e controle do projeto e da produção do software. Outros são documentações técnicas, incluindo o próprio código fonte do software (e em especial, os comentários contidos neste), forma bastante simples, produtiva e eficaz de documentação. Os artefatos-chave do RUP são comuns a todo tipo de software, sofrendo leves modificações de acordo com o domínio a que são aplicados.

4.3.1.1.1 Documento de Visão

O documento de visão apresenta uma visão geral do software a ser projetado e construído. Possui uma lista das suas principais características (*features*). Este artefato ajuda o time a entender o que deve ser construído, e também, a verificar se o que foi construído atende ao este conceito inicial. No processo de desenvolvimento de jogos, esse documento transforma-se na Especificação Conceitual do Jogo (*Conceptual Game Design*), que descreve

uma visão inicial do jogo a ser construído.

4.3.1.1.2 Lista de Riscos

Este artefato contém uma lista de todos os riscos detectados e quais as medidas de contenção e resolução que podem ser tomadas sobre cada um. Riscos são considerados sobre elementos do projeto: pessoas, processo e recursos (ferramentas de hardware e software). Riscos devem ser priorizados de acordo com a probabilidade de acontecerem e com os custos associados a eles. É importante que a lista esteja sempre atualizada para refletir os riscos reais do projeto.

4.3.1.1.3 Plano de Desenvolvimento

Descreve como o RUP está configurado para atender as necessidades do projeto. O Plano de Desenvolvimento contêm informações como as responsabilidades de cada papel e os Marcos Principais do projeto, e seus critérios de aceitação.

4.3.1.1.4 Casos de usos

Uma das características fundamentais do RUP é ser dirigido por casos de uso (seção 2.3.1.1). Como exemplo, os casos de uso servem como fundação para a definição dos requisitos do sistema e para o planejamento de testes. É possível usar casos de usos no desenvolvimento de jogos, embora a Especificação do Jogo tenha em jogos o mesmo papel que a Especificação de Casos de para um sistema de informação, guiando todo o processo de produção do jogo.

4.3.1.1.5 Plano de Testes

No RUP os testes podem ser planejados tão logo os primeiros Casos de usos forem identificados e documentados. Segundo [LARMAN 2001], o planejamento antecipado dos testes, baseado em casos de uso e especificação de requisitos, cria diretrizes sólidas para a implementação, aumentando as chances de construção de um software que satisfaça os requisitos almejados. Essa abordagem assemelha-se ao *Test-First Design*, prática bastante

difundida em XP. No desenvolvimento de jogos os testes são comumente planejados sobre a Especificação do Jogo.

4.3.1.1.6 Especificação da Arquitetura

É possível caracterizar duas formas de definição de uma arquitetura: durante a própria construção do software ou antecipadamente a etapa de implementação. Alguns times começam a implementação sem uma arquitetura definida e com o tempo ela é estabelecida. Outros definem a arquitetura inicialmente e somente depois, seguem com a implementação. Em ambos cenários, é possível que ocorra uma evolução da arquitetura. É fundamental existir a definição de uma arquitetura, mesmo que de forma paralela ou tardia à implementação.

No desenvolvimento de jogos, toda documentação técnica é apresentada na Especificação Técnica. Exemplos do conteúdo deste artefato são a descrição da arquitetura do software e possíveis soluções e padrões de codificações.

4.3.1.1.7 Plano de Projeto

O plano de projeto não sofre alterações para o desenvolvimento de jogos. Seu objetivo continua sendo o cronograma de desenvolvimento, planejamento das fases e iterações, planejamento de recursos e custos do projeto. A proposta do RUP é organizar as iterações de forma a mitigar riscos durante a fase de Elaboração. Desta forma, há diminuição da possibilidade de surpresas ou re-trabalhos posteriores.

4.3.1.1.8 Glossário (Dicionário de Termos do Projeto)

O glossário contém definições de termos usados no projeto. Seu objetivo é melhorar a comunicação dentro do time de desenvolvimento. Mesmo sendo um dos artefatos chave do RUP, ele é considerado opcional. Caso todos os participantes do projeto, incluindo interessados, entendam a linguagem do domínio, ele pode se tornar dispensável.

No desenvolvimento de jogos, o glossário pode ajudar novos desenvolvedores e interessados a entenderem a específica terminologia de jogos.

O glossário é um artefato de alta reutilização, pois boa parte dos termos pode ser reaplicada a outros projetos, tais como: *Level Design*, PC, NPC, *Items e* etc.

4.3.1.2 Processo Iterativo e Incremental

A criação e desenvolvimento de jogos são fortemente baseados em retornos. É improvável e difícil elaborar soluções que satisfaçam todos os seus requisitos no primeiro esforço do projeto. Os problemas são comumente melhor resolvidos de forma gradativa e incremental, o que é facilmente perceptível em softwares. Em jogos, o desenvolvimento interativo e incremental é essencial.

Como processo fortemente iterativo, a Criação de Jogos é responsável por dois quesitos qualitativos fundamentais: atração e diversão. Esses dois quesitos podem constituir a diferença entre um grande sucesso e um produto normal. Um jogo deve atrair e capturar jogadores para que eles queiram jogá-lo. A diversão garante a continuidade e a fidelidade do jogador ao jogo.

Garantir atratividade e diversão é uma tarefa altamente dependente de retornos de usuários reais, o que deve ser feito a todo o momento do ciclo de vida do jogo. Usuários de jogos comumente usados para testes em projetos são integrantes do próprio time ou jogadores do público-alvo. O processo deve permitir mudanças advindas dos retornos destes usuários.

A iteratividade está fortemente ligada ao desenvolvimento incremental. Ambos fazem parte das características fundamentais do RUP (seção 2.3.1.3). A conhecida expressão "*One mile wide and one inch deep*" ([KRUCHTEN 2000] e [JACOBSON et. al. 1999]) representa a visão do modelo iterativo incremental do RUP. Deve-se focar um planejamento amplo, detalhando as especificações à medida que os problemas passam a ser melhor conhecidos. Segundo [KROLL & KRUCHTEN 2003] e [POLLICE et. al. 2003], os problemas podem ser complexos demais para serem desvendados sem que nenhuma linha de código esteja escrita.

Em jogos, o desenvolvimento iterativo incremental é natural. Os conceitos do jogo e

os problemas do desenvolvimento do software são, em primeira instância, discutidos de forma ampla e superficial. Com a progressão das iterações, são realizados planejamentos que contêm detalhamentos gradativos do plano inicial.

No RUP, cada iteração pode ser considerada um mini-projeto em cascata, embora a ordem de execução das atividades não seja estrita. Cada iteração passa através das atividades de disciplinas do processo. As iterações podem ter duração fixa (*time-boxed*) ou um número fixo de recursos do jogo a serem implementados *(feature-boxed*).

4.3.1.3 Níveis de Formalidade

Os artefatos produzidos em um processo podem ser feitos sob diferentes níveis de formalidade. Quanto maior a necessidade de controle, maior a de formalidade. Em projetos complexos, com muitos desenvolvedores, é importante que os documentos sejam elaborados de acordo com um padrão da organização.

O nível de formalidade tende a diminuir em projetos ágeis, onde existe um foco maior no principal artefato do projeto, o software. É importante observar que a formalidade não influencia na qualidade do software produzido ([LARMAN 2001] e [POLLICE et. al. 2003]). A necessidade de maior formalidade está ligada ao número de integrantes da equipe, à complexidade do projeto e às exigências e padrões da organização executora (e cliente).

Dependendo da forma como o projeto é definido, o nível de formalidade pode variar: alguns documentos, especificamente internos, podem ser produzidos informalmente, enquanto outros, destinados ao cliente ou a outros envolvidos externos à produção, podem ser preparados formalmente.

O conceito de diferentes níveis de formalidade é suportado pelo Processo Unificado da Rational, que provê dois conjuntos de modelos (*templates*) de documentos: formal e informal.

No RUP Lite, a formalidade é modelada de acordo com o projeto ao qual o processo for aplicado. Um projeto de jogo 2D simples, com poucos desenvolvedores e de curta duração

pode não exigir alta formalidade, ao passo que um jogo 3D, certamente precisa de um controle maior sobre a documentação e comunicação.

4.3.1.4 Adoção de Práticas Ágeis

A *Extreme programming* possui, originalmente, doze práticas que se tornaram bem conhecidas ([BECK 2004]). Observadas ao longo do tempo em vários projetos, foram documentadas como boas práticas. [LARMAN 2001], [POLLICE et. al. 2003] e [KROLL & KRUCHTEN 2003] apóiam o uso de algumas práticas de XP no Processo Unificado da Rational. O objetivo desta sessão não é apresentar as práticas XP, mas sim, a forma como elas são enquadradas no RUP Lite e no desenvolvimento de jogos móveis.

Segundo autores da própria Rational, tais como [LARMAN 2001], [POLLICE et. al. 2003] e [KROLL & KRUCHTEN 2003], não é necessário o cenário RUP seja leve para compatibilizar a adoção das práticas ágeis. Essas práticas são chamadas de ágeis por serem enfatizadas em metodologias como XP, embora sejam uma compilação de práticas existentes antes do termo “ágil” ser usado. Esta é também uma razão pela qual tais práticas podem ser implementadas pelo RUP.

Das doze práticas apresentadas em XP, algumas são discutidas dentro do cenário de um RUP Lite – Simplicidade, Re-fabricação, Jogo de Planejamento, Integração Contínua, Pequenos Lançamentos e Cliente no Local. Outras são adaptadas para melhor atender ao desenvolvimento de jogos com RUP – Programação em Pares e Propriedade Coletiva.

4.3.1.4.1 Programação em Pares “Critica” (“Critical” Pair Programming)

Sugere-se usar a prática de programação em pares somente nas partes mais críticas (complexas ou de alto risco) do software. O trabalho em pares em tempo total pode, em alguns casos, causar diminuição da produtividade, pois existem problemas conhecidos quanto a esta prática ([LARMAN 2001], [LARMAN 2003]). Nem todos os desenvolvedores conseguem trabalhar desta forma.

Por um outro lado, o trabalho em pares é excelente para revisões de código em tempo de criação. Sem o uso integral desta prática, sugere-se realizar revisões semanais em duplas, e uma ou mais inspeções formais de código por lançamento.

4.3.1.4.2 Propriedade Coletiva (Colective Ownership)

A programação em pares “crítica” ou “momentânea” permite que parte do código seja coletivo. Geralmente, as partes mais simples de um código podem ser alteradas por outros desenvolvedores sem grandes problemas. São as partes críticas que merecem maior atenção e muitas vezes tornam um desenvolvedor proprietário de um código. Desta forma, existe razão para o uso do trabalho em pares em partes críticas – se adicionado o benefício da propriedade coletiva –, mesmo que de forma diferente da originalmente concebida em XP. Ao menos dois desenvolvedores estarão aptos a modificar e responder por um código considerado crítico do software.

4.3.1.4.3 Simplicidade (Simplicity)

Em grande parte dos casos, a solução mais simples é a melhor. É preferível usar soluções simples, evitando tentativas de "previsões" de reutilização futura. Assim, deve-se planejar o software cuidadosamente para saber quando construir classes e componentes genéricos. É importe planejar para reutilizações futuras, mas é preciso possuir certeza desta reutilização.

Manter não só o código, como também soluções e artefatos em sua forma mais simples ajuda a agilizar a produção e a resposta a mudanças. Usa-se menos tempo e esforço à medida que não serão projetadas soluções altamente genéricas e, conseqüentemente, mais complexas.

A simplicidade no projeto e codificação, e nas soluções algorítmicas ajuda a manter a prática de propriedade coletiva, pois a maior parte do código provavelmente estará em sua forma mais simples (para entender e modificar).

A simplicidade também deve estar fortemente presente na Criação de jogos do jogo. Colocar mais recursos ou aumentar a complexidade das já existentes não garante um bom *gameplay*. Enganosamente relaciona-se ao bom *gameplay* a uma grande quantidade de recursos. Um jogo deve ser sempre pensado sob a ótica da prática "max-min": tirar o máximo de jogabilidade de um número mínimo de recursos ([PEDERSEN 2003]).

Assim como os artefatos e o processo, a criação de jogos também precisa ser iterativa e incremental. Considerada atividade evolutiva, é preciso garantir o *gameplay* por testes e retornos (Seção 3.3). Portanto, trabalhar com a forma mais simples traz bons resultados.

Existe uma regra simples para a evolução do conceito de um jogo:

"Primeiramente, é definido um conceito simples. A partir disto, um protótipo é criado e experimentado várias vezes. Só então, consegue-se chegar em um conceito de jogo mais estável (calibrado). A partir disto, surgem idéias. Com o *gameplay* em sua forma mais simples e calibrada pode-se então acrescentar novos recursos, caso seja necessário. Desta forma, é possível agregar valor ao jogo gradativamente, ao invés de apenas acrescentar complexidade e riscos ao projeto à medida que são incluídos recursos em um *gameplay* instável. Deve-se sempre prezar por um *gameplay* estável, consistente e simples".

4.3.1.4.4 Re-fabricação (Refactoring)

Para sempre tentar manter a forma mais simples, e conseqüentemente um bom projeto de código, é preciso fazer re-fabricação. É importante re-fabricar o código para deixá-lo em uma forma mais simples (quando possível ou necessário). Por outro lado, não se deve aplicar uma re-fabricação excessiva, pois desta forma, o tempo ganho com a simplicidade é perdido com o excesso de re-trabalhos. A chave é o bom senso dos desenvolvedores.

A simplicidade e a re-fabricação são práticas também importantes para diagramas e especificações, tais como a Especificação do jogo e a Especificação Técnica.

4.3.1.4.5 Jogo de Planejamento (Planning Game)

Em concordância com os fundamentais iterativos e incrementais do RUP ("*A Mile Wide and an Inch Deep*" ([KRUCHTEN 2000] e [JACOBSON et. al. 1999]), um planejamento amplo e superficial é realizado inicialmente. Com o progresso das iterações, cada parte do software é descrita em mais detalhes e implementada (mini-projetos).

No desenvolvimento de jogos, a produção da Especificação do Jogo é uma das atividades que usa esta prática: uma visão geral é gerada pelo Líder de Criação de jogos, que divide o jogo em partes a serem detalhadas pelos demais Criadores de Jogos da equipe. A criação dos níveis do jogo também usa esta prática, pois eles são detalhados somente na etapa de Criação de Níveis do processo. Antes disso, existe apenas uma descrição em alto nível dos requisitos e aparência dos níveis.

4.3.1.4.6 Integração Contínua (Continuous Integration)

A produção de jogos é um processo de desenvolvimento modular: personagens podem ser implementados em quase total isolamento dos cenários. No entanto, é preciso garantir que as interações entre eles funcionem corretamente. A prática de integração contínua oferece bons resultados e diminui de forma significativa os riscos de finalizar o projeto com um "todo que não condiz com a soma de suas partes". Do ponto de vista da criação de jogos, essa integração diária permite visualizar como todos os elementos do jogo estão interagindo e de que forma estão afetando o *gameplay*.

4.3.1.4.7 Pequenos lançamentos (Small Releases)

É importante que haja lançamentos (*releases*) internos ou externos regularmente. Testes contínuos de jogabilidade realizados sobre lançamentos intermediários podem ajudar a garantir que o jogo satisfaça as expectativas funcionais e de jogabilidade.

4.3.1.4.8 Cliente no local (On-Site Customer)

O desenvolvimento de jogos é uma área onde, naturalmente, o "cliente" está no local. O Criador de Jogos é o papel que faz as exigências para os requisitos do software. Ele deve

estar sempre presente, pois é de sua responsabilidade garantir o foco do projeto. Em outras palavras, ele precisa assegurar que a idéia do jogo está sendo compreendida e mantida ao longo do desenvolvimento por todos os integrantes da equipe.

4.4 Sumário

Este capítulo apresentou um “elo” ou ligação entre os Capítulos 2 (*Rational Unfied Process*) e 3 (Desenvolvimento de Jogos), e o Capítulo 5 (*Rational Unfied Process para Jogos Móveis*). Foram apresentadas as características e peculiaridades de projetos de jogos móveis a fim de demonstrar o que é necessário para um processo se tornar mais adequado a este domínio. Desta forma, tem-se o RUP Lite, processo que exige e enfatiza as características fundamentais do RUP. Tal processo serve de base para o MGUP (Processo Unificado para Jogos Móveis), apresentado no próximo Capítulo.

5 Processo Unificado da Rational para Jogos Móveis

5.1 Introdução

A discussão deste capítulo fundamenta-se nos anteriores para apresentar uma discussão detalhada da aplicação do Processo Unificado da Rational no desenvolvimento de jogos móveis. Essa aplicação será referenciada como MGUP (Processo Unificado para Jogos Móveis).

O MGUP toma por base o RUP Lite apresentado na seção 4.3 e representa uma fusão entre os assuntos discutidos nos capítulos 2 (RUP), 3 (Desenvolvimento de Jogos) e 4 (RUP e Jogos de Celular).

5.2 Necessidades atuais dos jogos móveis

Na época em que os jogos eram pequenas aplicações, uma discussão em torno de desenvolvimento de jogos móveis seria direcionada para um processo absolutamente ágil provavelmente baseado na disciplina *Extreme programming*. Limitados pelos recursos escassos dos dispositivos da época, os jogos precisavam ser bastante simples. Em conseqüência, os perfis da grande maioria dos projetos de jogos móveis da época eram de projetos de curta duração (até três meses) e se equiparavam em complexidade a jogos mais simples da plataforma Nintendo NES. Hoje, essa realidade mudou.

Atualmente, os jogos 2D desenvolvidos com as tecnologias J2ME e Brew estão perto de títulos complexos de plataformas como Sega Genesis e Nintendo SNES. Vários jogos já são equivalentes em complexidade aos projetos mais sofisticados destas plataformas. Além disso, estão surgindo cada vez mais jogos 3D, com qualidade visual equiparável à plataforma Sony Playstation. Embora o *gameplay* dos jogos de celular seja em sua maioria simples, o potencial gráfico e sonoro de aparelhos cada vez mais sofisticados têm permitido que

desenvolvedores produzam jogos cada vez mais requintados. Conseqüentemente, a complexidade dos projetos tem aumentado gradativamente, evoluindo de forma alternativa aos jogos criados para plataformas mais tradicionais. Os jogos para consoles e PC's evoluíram tanto em recursos multimídia quanto em jogabilidade (tornando-se eventualmente mais complexa). Jogos móveis mantêm o mesmo padrão de evolução multimídia já visto nestas plataformas tradicionais, mas o *gameplay* está tomando um caminho diferente, indo para um modelo mais simplista. Uma provável razão para este curso de evolução é o público alvo dos jogos móveis: jogadores casuais que não têm interesse em um *gameplay* complexo.

Neste cenário de desenvolvimento de jogos cada vez mais complexos, a agilidade precisa ser mantida, mas um espaço para maior controle e formalismo deve estar aberto. Essa é a linha guia para a discussão a ser realizada sobre a aplicação de um processo como RUP para o desenvolvimento de jogos móveis.

Dando seqüência ao capítulo anterior, a seguinte discussão parte de um cenário de RUP já apresentado: o RUP Lite. Esse cenário inicial foi usado para facilitar a inclusão, modificação ou remoção de elementos do processo para melhor se adequar ao domínio de jogos sem ficar preso a detalhes específicos do RUP, como modelos de documentos formais ou descrições burocráticas de papéis e atividades.

5.3 Princípios do MGUP

O RUP possui três princípios fundamentais (também chamados características fundamentais). Esses princípios levam em conta características de processos aplicados de forma mais genérica, podendo ser estendidos a todo tipo de software. A aplicação do RUP para jogos móveis parte destas características fundamentais, não sendo somente uma configuração (*tailoring*) do processo.

5.3.1 Dirigido à Criação de Jogos

O cenário de aplicação do RUP apresentado não usa o modelo de Casos de Uso para

guiar o desenvolvimento. O modelo de casos de uso descreve interações de usuários com o sistema de forma seqüencial, apresentando também fluxos alternativos. Sistemas não iterativos podem ser descritos facilmente por meio desta técnica. No entanto, sistemas multimídia que funcionam de forma iterativa não executam regras de forma seqüencial. Sua descrição por meio de casos de uso tende a tornar-se mais difícil de ser realizada.

Sabe-se que na maioria dos casos, a concepção de jogos não é feita utilizando-se casos de usos como ferramenta de descrição. Ao contrário, uma outra forma de documentação mais flexível é usada: o documento de Criação de jogos. As especificações de criação de jogos assumem no RUP as mesmas funções dos documentos de casos de uso, tornando-se chave para o desenvolvimento de um jogo.

Como visto no capítulo 3, a criação de jogos possui função primordial na produção de um jogo. Ele descreve a visão inicial (Especificação conceitual do jogo), contendo os recursos mais importantes e características gerais; e, descreve uma visão detalhada do software (Especificação do jogo), que é usada como base para o planejamento de requisitos, projeto de software e arquitetura. Além disso, todos os testes funcionais, não-funcionais e de jogabilidade (e usabilidade) são planejados e executados com base nos documentos de criação de jogos. Pode-se dizer que o processo de garantia de qualidade é guiado pelo conceito do jogo. Desta forma, fica mais que claro que um processo de jogos é totalmente dirigido pela criação de jogos.

5.3.2 Processo Iterativo e Incremental

A tarefa de concepção de um jogo não é ainda uma ciência. Criar o conceito de um jogo que seja interessante ao jogador depende fundamentalmente dos testes e retornos, que são realizados durante todas as etapas de produção do software. Estes retornos conseqüentemente, geram necessidade de mudanças.

De acordo com [BETHKE 2003], projetos de jogos possuem escopo que se modifica

ao longo do ciclo de desenvolvimento. Essas mudanças tendem a ocorrer em escalas decrescentes. Em boa parte dos casos, as mudanças são causadas por alterações do conceito do jogo, que está preocupado em manter a qualidade do jogo. Outros fatores que levam a mudanças no jogo são limitações técnicas da equipe e do dispositivo no qual se está trabalhando. Faz parte do cotidiano do desenvolvimento de jogos a detecção de novos problemas que necessitem de algoritmos mais sofisticados como solução. Algumas soluções concebidas pelo Criador de Jogos podem não ser possíveis de serem implementadas no dispositivo e algumas vezes não é possível certificar-se dessa limitação antes da implementação. Além disso, fatores como prazo e tamanho de equipe também causam modificações no conceito do jogo.

As mudanças dentro do universo de criação de jogos são tão comuns que é possível um jogo acabar bastante diferente de como foi originalmente concebido ([BETHKE 2003]). Obviamente, existe um fechamento à ocorrência e ao grau dessas mudanças ao longo do ciclo (sessão 5.4.2.1).

O processo de criação, de uma forma geral, não é linear e discreto. Pinturas, músicas, automóveis e jogos, são alguns exemplos de produtos que envolvem um alto esforço criativo. Na criação, não existe fórmula ou caminho pré-determinado: é preciso experimentar ([BETHKE 2003]). Obviamente, a experiência dos participantes da equipe ajuda na delineação de caminhos mais prováveis de sucesso, mas ainda assim, iteratividade, desenvolvimento incremental e retornos são ferramentas chave para todo processo de criação. Logo, observa-se que um processo de desenvolvimento de jogos eficiente deve ser iterativo e incremental.

Na criação de jogos, é bastante difícil conceber um jogo interessante sem esforços consideráveis. Seguir um processo em cascata, que determina que todo o jogo deva ser especificado no início do ciclo e só então depois implementado por completo, pode criar muitas barreiras à execução do processo natural de criação. Processos em cascata tentam

minimizar o re-trabalho dificultando a realização de mudanças nos artefatos produzidos em etapas anteriores. Em um jogo, isso é essencial para se modelar a forma como ele é jogado: a chamado *gameplay*.

5.3.3 Centrado na Arquitetura e no Core gameplay

O RUP não exige a utilização de diagramas de pacotes ou componentes, mas é centrado em arquitetura e em sua essência, prioriza a criação de uma arquitetura executável logo cedo, assim como, a construção do sistema em componentes, reforçando a reutilização. O MGUP mantém todas essas características.

O *core gameplay* define o que é mais importante no jogo. A arquitetura inicial do software, por sua vez, deve permitir que os elementos do *core gameplay* sejam implementados. Dentro do processo de desenvolvimento de um jogo, a primeira etapa de produção é centrada na criação de uma arquitetura inicial executável – etapa de Criação de Componentes. Além disso, a etapa de Produção do Jogo possui foco no *core gameplay*. Desta forma, a primeira versão executável deve refletir o *core gameplay* e possuir os principais recursos arquiteturais já implementados.

A arquitetura de software é descrita como diferentes visões do sistema sendo construído. O conceito de arquitetura envolve os aspectos estáticos e dinâmicos mais significativos do sistema. Desta forma, pode-se dizer que a descrição do *core gameplay* é a primeira versão da arquitetura do jogo, pois contém os elementos primordiais à sua existência.

De acordo com o apresentado na seção 2.3.1.2, no MGUP, assim como no RUP, o artefato de Especificação do Jogo (substituto dos casos de uso) e a arquitetura estão relacionados, pois todo jogo deve ter tanto função, quanto forma. Esses dois aspectos precisam ser balanceados para alcançar um jogo de sucesso, uma vez que os elementos do conceito do jogo precisam estar presentes por meio do software e ao mesmo tempo, para que isso seja possível, a arquitetura deve suportar a criação destes. Desta forma, a arquitetura deve

ser construída em uma seqüência lógica, que está diretamente relacionada com o detalhamento e evolução do conceito do jogo e, especialmente, do *core gameplay*.

5.4 MGUP: Processo Unificado para Jogos Móveis

A discussão sobre a aplicação do Processo Unificado da Rational no desenvolvimento de jogos móveis é feita sob o mesmo formato pelo qual foi apresentado o próprio RUP, no capítulo 2. Esse formato apresenta o processo em termos dos elementos de estrutura dinâmica e estrutura estática.

5.4.1 Estruturas Dinâmica e Estática do MGUP

Seguindo o formato tradicional, uma aplicação do RUP para jogos móveis pode também ser descrita em termos das estruturas (ou dimensões) do processo: estrutura dinâmica e estrutura estática (Figura 6).

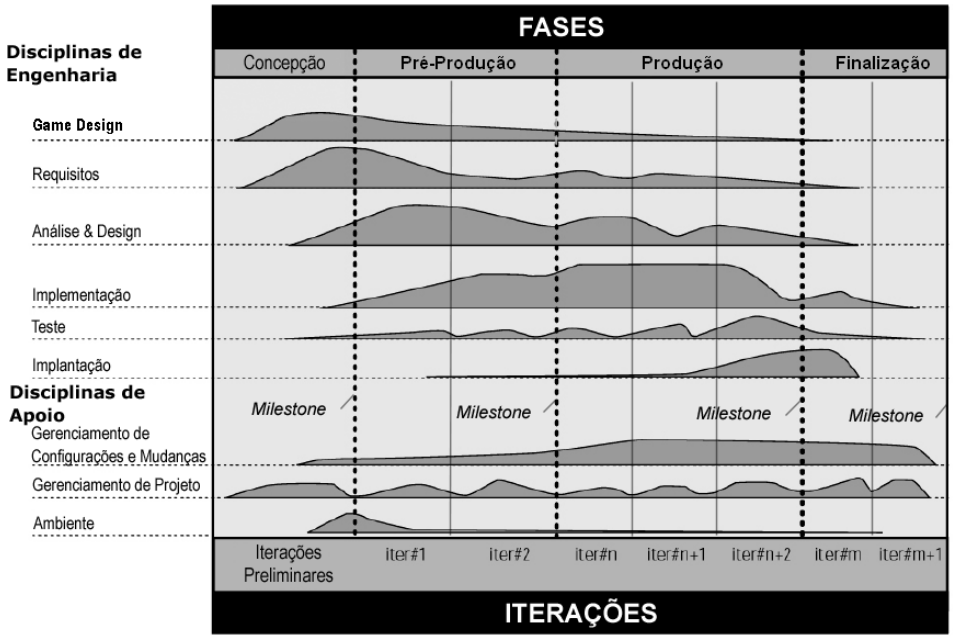

Figura 6: Estrutura de dimensões disciplinas e esforço do MGUP.

A estrutura em duas dimensões mostra a estrutura dinâmica (eixo horizontal) e a estática (eixo vertical) do MGUP. No diagrama observa-se a substituição da disciplina de Modelagem de Negócio pela disciplina de Criação de jogos. Também, estão apresentadas as

quatro fases seqüenciais do processo, presentes da mesma forma que a estão no RUP, apenas com uma nomenclatura mais adequada a área de desenvolvimento de jogos.

5.4.2 Estrutura Dinâmica do processo

O MUGP possui quatro fases distintas que evoluem à medida que se satisfazem seus Marcos Principais. A estrutura dinâmica segue o mesmo formato do RUP, sendo expressa em termos de ciclos, fases e marcos. As quatro fases possuem marcos definidos para que se possa mensurar a progressão entre elas ao final de cada uma.

O processo de produção de jogos, de forma geral, possui as mesmas características que o desenvolvimento de outros tipos de software. O desenvolvimento interativo é fortemente enfatizado e existe uma seqüência de fases lógicas que determinam os Marcos Principais de um projeto de jogos. Baseando-se no modelo de fases apresentado no capítulo 3, pôde-se descrever a produção de um jogo como um processo dividido em quatro fases, análogas às encontradas no RUP. Essa configuração de fases e a descrição de cada uma apresenta um modelo genérico de produção de jogos, descrito com base em relatos de projetos reais. Cada projeto deve adaptar o modelo à sua realidade.

O MGUP é centrado na produção de artefatos de arte gráfica, mídia sonora, software e conceitos do jogo. Pode-se dizer que um processo de jogos é dirigido pela criação de jogos, visto que todas as especificações, planejamentos, implementações e testes são feitos baseando-se neste documento.

Espera-se que com o decorrer do ciclo do projeto, o número e a complexidade (grau de impacto) de mudanças diminuam. Mudanças são normais e tanto o processo quanto a equipe devem estar preparados para absorvê-las. No entanto, dificilmente tem-se um projeto onde as mudanças podem ser constantes em qualquer número e complexidade.

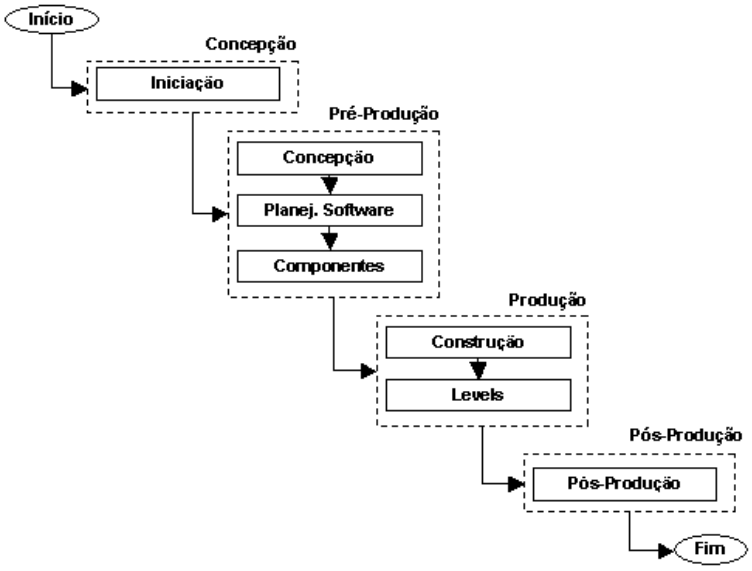

Diagrama 4: Organização das fases de jogos para o MGUP.

As fases genéricas do *Game Process* apresentadas no capítulo 3 foram agrupas e distribuídas de acordo com as etapas macro de desenvolvimento de um jogo (Diagrama 4):

- **Concepção**: É a etapa inicial do processo e representa a parte contratual e de definições inicias do projeto. Aqui são realizados: concepção da idéia geral do jogo, descrição dos principais recursos, fechamento de contrato, e planejamento inicial do projeto (fases, Marcos Principais, pessoas, papéis, artefatos, custos e recursos).
- **Pré-Produção**: Compreende a etapa de planejamento de todo o jogo. Além do conceito do jogo (*game design*), o planejamento se estende de forma completa a todas as outras características do jogo: software, arte e som. Esse é o momento onde todo o jogo é concebido. Vários testes com animações-exemplo, imagens conceituais e protótipos jogáveis são realizados a fim de que se possa planejar o jogo da forma mais precisa possível. É claro que sempre sobram alguns pontos, especialmente os detalhes muito pequenos (ex.: qual a velocidade em *pixels* por *frame* do personagem?). Essa é sem dúvida a etapa mais crítica do projeto. Um planejamento mal elaborado ou pouco detalhado pode fazer com que o jogo perca o foco e se transforme em algo diferente

do que foi inicialmente concebido pelo Criador de Jogos.

- **Produção**: Compreende a etapa mais longa do processo, sendo geralmente composta por várias iterações. É também o momento no qual o projeto passa a ter o maior número de pessoas. A produção do jogo segue tudo o que foi planejado na fase anterior. Por esta razão é que se deve fazer terminar a Pré-Produção com o planejamento mais minucioso possível.
- **Pós-Produção**: A etapa final do processo compreende todas as atividades de acabamento do jogo. A equipe de qualidade entra em cena e muitos testes são realizados, em especial os que têm mais relação com os usuários finais: testes funcionais e testes de jogabilidade. Modificações e pequenos ajustes são realizados para se corrigir problemas encontrados e fazer pequenas melhorias.

5.4.2.1 A Dimensão de Mudanças

O processo de desenvolvimento está dividido em quatro etapas. Considera-se que mudanças possam ocorrer durante o desenvolvimento do projeto. No entanto, entende-se que o nível e a quantidade destas diminuam no decorrer do ciclo. A razão é simples: mudanças críticas no final ou mesmo em uma fase inicial, mas decisiva do projeto, pode comprometer o sucesso. As mudanças no conceito do jogo são as que mais ocorrem durante a produção de um jogo e podem ser as de maior risco.

O diagrama clássico do RUP (Figura 6) reflete a dimensão do esforço que é aplicado em cada disciplina ao longo do ciclo de desenvolvimento. A Figura 7 e a Figura 8 refletem uma outra dimensão do MGUP: as mudanças ao longo do ciclo (Fases x Disciplinas x Mudanças). Como já discutido na seção 3, as mudanças têm um papel essencial na evolução do conceito do jogo, conseqüentemente, em sua qualidade.

A Figura 7 reflete a aceitação do processo a mudanças no escopo do projeto. No início, é possível que ocorra, praticamente, qualquer tipo de mudanças. Da fase de Concepção

até o final do projeto, o leque abertura a mudanças vai se fechando. Ao final, é altamente arriscado se permitir mudanças na fase de Finalização. Esse é um momento apenas para ajustes no software.

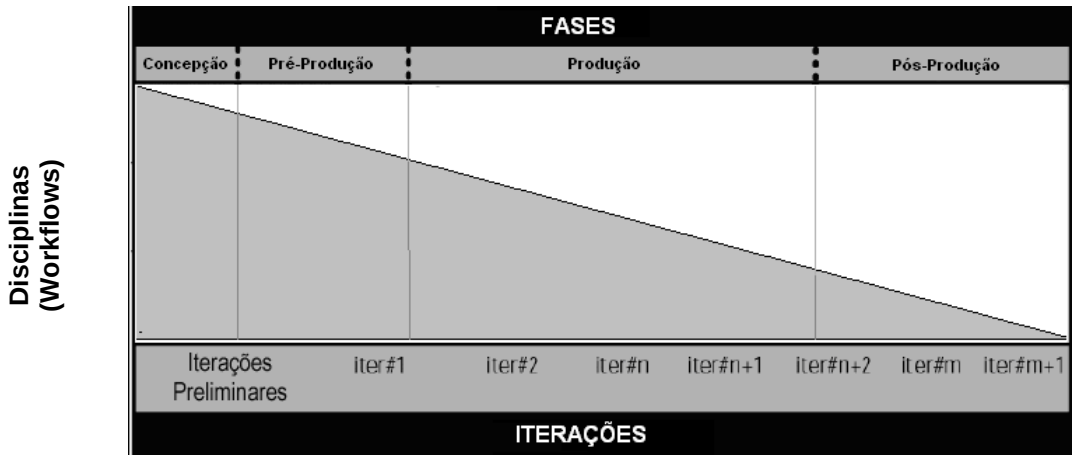

Figura 7: Dimensão: "Fechamento" do processo a mudanças.

Na Figura 8, observa-se uma outra dimensão das mudanças no projeto. Na fase de Concepção, as mudanças são bastante conceituais, pois não é o momento onde se tem uma visão detalhada do jogo e nem testes de tecnologia. Pode-se mudar praticamente qualquer elemento do jogo, incluindo seu próprio conceito. Nesse momento, ainda se tem uma visão geral do jogo e de seus principais recursos. Na fase de Pré-Produção, ocorre o maior número de mudanças, pois este é o momento em que o jogo é definido em cada detalhe, para posteriormente, na fase de Produção, ser construído por completo. Essas mudanças possuem alto impacto em todo o projeto. Em outras palavras, a etapa de Pré-Produção é o momento em que o jogo é criado. Depois dela, podem ocorrer algumas melhorias, mas o jogo será o que for definido nesta fase. È importante que testes de jogabilidade sejam realizados sobre os protótipos construídos para que seja possível realizar mudanças bastante significativas já no conceito do jogo.

Na Produção, mudanças de impacto médio ainda são aceitas. Neste momento, modificações no conceito do jogo ou em seus elementos podem ocorrer, mas de forma que não exijam refazer a *engine* do jogo. Nesse momento, pode-se estar ainda discutindo o acréscimo de um nível ao jogo, uma nova habilidade ao personagem, o número de inimigos a

serem criados. No decorrer desta fase, as mudanças devem diminuir em complexidade e número. Na fase de Finalização, não devem existir mudanças no jogo, mas sim, pequenos ajustes, principalmente devido à parte de testes.

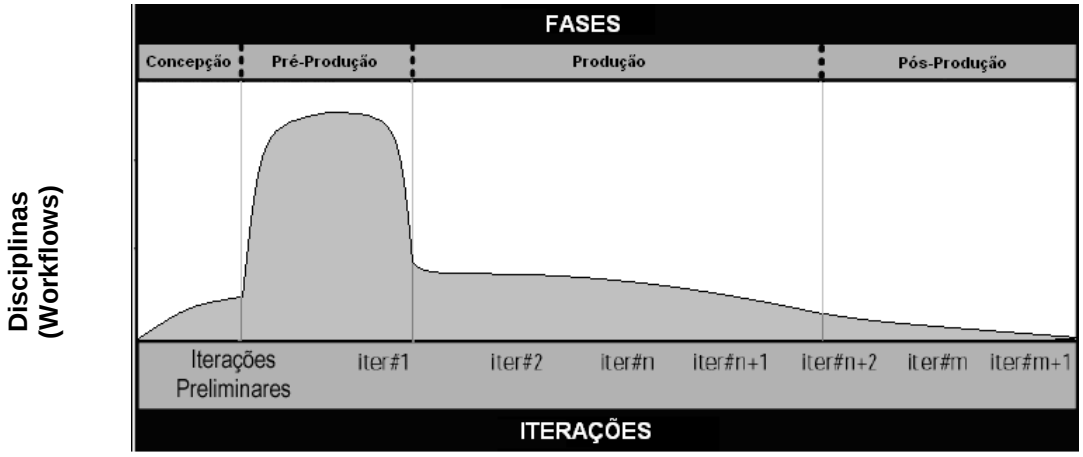

Figura 8: Dimensão: Nível de importância das mudanças.

Quanto à produção de protótipos, a Pré-Produção apresenta-se como momento ideal para testes, experimentos e provas de conceito. Da tecnologia ao conceito do jogo, protótipos podem ser criados (e comumente isto é feito) para se validar e garantir que ao final da fase, se tem uma idéia exata de como será o jogo.

Obviamente é comum a realidade diferir de um projeto ideal. É possível que ocorram mudanças em momentos inadequados como, por exemplo, mudanças de especificação do jogo na fase de Produção. No entanto, o ideal é não considerar isto uma prática comum e sempre trabalhar para que isto ocorra cada vez menos à medida que novos projetos são realizados.

5.4.3 Estrutura Estática do Processo

A abordagem descritiva utilizada no MGUP é baseada em apontar as diferenças em relação ao RUP: o que precisa ser acrescentado, modificado ou removido para que ele se torne mais adequado ao desenvolvimento de jogos móveis. Sendo assim, elementos do RUP que não sofrem alterações não são abordados, pois isso teria tornado a investigação um tanto quanto redundante.

As mudanças do MGUP em relação ao o RUP estão bastante centradas ao redor do

conceito do jogo. Pode-se dizer que a criação de jogos é uma atividade absolutamente específica da produção de jogos. A descrição das modificações sobre o RUP está direcionada aos elementos que fazem parte das estruturas dinâmica e estática do RUP: fases, disciplinas, papéis, atividades e artefatos.

- **Fases**: O MGUP segue o mesmo modelo de divisão do processo em quatro fases do RUP: Concepção, Elaboração, Construção e Transição. A nomenclatura é modificada para melhor se enquadrar ao domínio de jogos. Assim, as quatro fases passam a se chamar Concepção, Pré-Produção, Produção e Finalização. Essa nomenclatura também acaba refletindo a usada na produção de filmes.
- **Disciplinas (Workflows):** As disciplinas do RUP são todas mantidas, com exceção da Modelagem de Negócios. A Modelagem de Negócios possui papéis, artefatos e atividades que não são adequadas ao domínio de jogos. Para uma aplicação do RUP em jogos, é proposto substituir essa disciplina por uma que apresenta maior adequação dentro do domínio de jogos: a disciplina de Criação de jogos.
- **Papéis:** O MGUP possui basicamente a mesma configuração de papéis do RUP. As diferenças estão na inclusão dos papéis específicos para jogos, tais como Artistas (2D, conceitual, 3D, animador, etc.) e Sonoplastas (músicas e efeitos sonoros); e também, na remoção dos papéis específicos da disciplina de Modelagem de Negócios, substituída por Criação de jogos.
- **Atividades e Artefatos**: Em conseqüência à inclusão de novos papéis referentes à arte, Mídia Sonora e criação de jogos, as atividades relativas a esses papéis são incluídas no MGUP, assim como os artefatos que são produzidos. Também, existem alterações na disciplina de testes para contemplar testes de jogabilidade.

5.4.4 Fases do MGUP

O MGUP possui quatro fases que compreende uma compilação do modelo RUP e do

Game Process genérico. É importante ressaltar que o Criador de Jogos sempre segue o Ciclo Iterativo (seção 3.3.1) para evoluir o conceito do jogo em qualquer fase do processo.

5.4.4.1 Concepção

A fase de Concepção do MGUP equivale à fase de Iniciação do Processo de Jogos apresentado na seção 3.4.2. No RUP, é preparado um protótipo de software nesse momento (seção 2.3.4.1), mas no MGUP isso é feito na fase de Pré-Produção, em concordância com o Processo de Jogos e com a ordem de atividades da evolução do conceito do jogo (sessão 3.3). A Especificação Conceitual do Jogo assume as funções dos documentos de Visão e Casos de Uso inicial.

O propósito geral desta fase é compreender o escopo do projeto. Com isso, produz-se um documento de visão (*Conceptual Game Design*); realiza-se um planejamento inicial do projeto (Plano de Projeto, Análise de Riscos e Plano de Desenvolvimento); e, consegue-se aprovação dos interessados (clientes) no jogo para prosseguir com o projeto. Segundo [FULLERTON 2004], a aprovação de um projeto de jogos na fase de Concepção é feita com base em três características: as pessoas do projeto, plano de projeto e a idéia do jogo.

Esta fase compreende as três primeiras etapas do processo de criação de jogos (*Brainstorming*, Protótipo Físico e Apresentação - Seção 3.3.2). Assim, pode ser preparado um protótipo físico para validação do conceito inicial do jogo descrito na Especificação Conceitual do Jogo. O uso ou não do Protótipo Físico está ligado a variáveis como: tamanho do projeto, complexidade do jogo, prazo e custo de produção destes documentos.

Resumo da Fase de Concepção

Duração: 3% do tempo de projeto.

Objetivos:

- Compreensão do escopo do projeto.
- Criação do Conceito Inicial (Especificação conceitual do jogo, com foco em marketing e diferenciais da idéia).
- Estudos iniciais de viabilidade.
- Construção do artefato Plano de Desenvolvimento.
- Aprovação dos interessados para prosseguir com o projeto.
- Produção de um glossário de projeto inicial.
- Preparação de um protótipo físico para validação do conceito inicial (opcional).

Papéis Envolvidos:

- Produtor.
- Líder de Criação de Jogos.
- Líder de Arte.
- Planejador de Software.
- Profissional do Departamento de Publicidade e Propaganda.
- Profissional do Departamento Comercial.

Resultados Esperados:

- Contrato.
- Especificação Conceitual do Jogo (e algumas Artes Conceituais).
- Plano do Projeto.
- Glossário do Projeto Inicial.
- Protótipo Físico (opcional).

Marco Principal: Objetivos do Ciclo de Vida (*Lifecycle Objectives Milestones – LCO*)

- Objetivos do ciclo de vida devem estar bem definidos.

5.4.4.2 Pré-Produção

Os objetivos desta fase são: produzir a concepção do jogo de forma completa, gerar uma documentação técnica detalhada, conceber o alicerce de produção do jogo (arquitetura executável) e construir um ou mais protótipos a fim de se validar e viabilizar elementos apresentados no documento de criação de jogos.

Em projetos de jogos típicos, o time envolvido neste primeiro protótipo é geralmente menor que o time completo de produção. Isso acontece porque o financiador não tem interesse em custear a equipe toda em um estágio preliminar do desenvolvimento. Esse

pequeno time inicial é responsável por produzir os documentos de planejamento do jogo: Especificação do jogo, Especificação Técnica e Especificação de Arte e Mídia.

A análise de viabilidades realizada neste momento serve para detectar problemas de alto risco. É preferível que um recurso de alto risco do jogo seja cortado nesta fase do que durante sua produção, onde o impacto de uma mudança pode ser grande.

O objetivo do protótipo em software construído nesta fase é ajudar a validar os conceitos do jogo. Também serve para garantir a aceitabilidade pelo financiador (às vezes pelos usuários finais também). Ele contém os principais elementos do *gameplay* (*core gameplay*). Geralmente, no modelo de negócios de jogos, esta é a etapa onde a publicadora averigua a capacidade e a viabilidade do jogo apresentado.

A fase de Pré-Produção é considerada a mais crítica no processo, uma vez que neste momento planeja-se o caminho a ser seguido para toda a produção do jogo. Quanto melhor produzidos os documentos de especificações do jogo, menor é a probabilidade de ser deparar com problemas posteriormente.

Esta fase do MGUP é equivalente às fases de Concepção do Jogo, Planejamento de Software e Construção de Componentes do modelo *Game Process* da seção 3.4.2 (Figura 9). Conseqüentemente, ela compreende as etapas de Protótipo de Software e Especificação do Jogo do processo de Criação de jogos (Seção 3.3.2).

A duração e o número de atividades a serem realizadas nesta etapa fazem com que no MGUP ela seja dividida em três macro-iterações: Planejamento do jogo, Planejamento do Software e Arte e Construção da Arquitetura (Figura 9). O Planejamento do Software e de Arte são feitos em paralelo. Dentro de cada macro-iteração pode haver mais iterações.

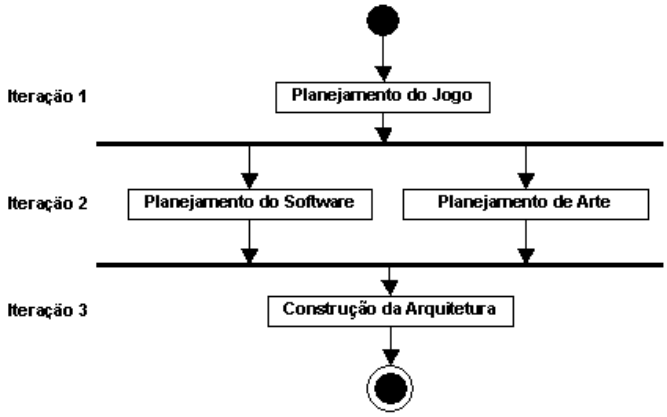

Figura 9: Fluxo da fase de Pré-Produção: 3 macro-iterações.

Resumo da Fase de Pré-Produção

Macro-Iteração 1: Planejamento do Jogo

Duração: 10% do tempo de projeto.

Atividades:

- Planejamento final do projeto.
- Planejamento do jogo: criação da Especificação do jogo.
- Verificação de Viabilidade de forma mais profunda.
- Validação dos conceitos com prototipação em software.
- Execução das etapas de Protótipo de Software e Especificação do Jogo do processo de Criação de jogos (Seção 3.3.2).

Papéis Envolvidos:

- Toda equipe de Criação de Jogos.
- Líder de Desenvolvimento.
- Arquiteto de Software.
- Planejador de Software.
- Líder de Arte.
- Artista Conceitual.
- Produtor.

Resultados:

- Especificação do jogo.
- Plano de Projeto Final.
- Protótipo de Software.

Macro-Iteração 2: Planejamento do Software

Duração: 7% do tempo de projeto (em paralelo).

Atividades:

- Avaliação das soluções necessárias para os recursos do jogo baseando-se na Especificação do jogo.
- Captura de Requisitos. Esta atividade (e seu artefato) é considerada opcional para alguns jogos. Projetos menores podem se basear diretamente pela Especificação do jogo.
- Planejamento do Software (arquitetura, soluções e Marco Principal).
- Mitigação de riscos técnicos de maior impacto.

Papéis Envolvidos:

- Líder de Criação de Jogos.
- Arquiteto de Software.
- Planejador de Software.
- Desenvolvedores de Software mais experientes.
- Líder de Arte.

Resultados:

- Documento de Requisitos (Opcional).
- Especificação de Software, contendo soluções e arquitetura inicial do software.
- Manual de boas práticas e padrões de programação.

Macro-Iteração 2: Planejamento de Arte

Duração: 7% do tempo de projeto (em paralelo).

Atividades:

- Planejamento da Arte e Mídia (incluindo Marco Principal).

Papéis Envolvidos:

- Líder de Criação de Jogos.
- Artistas Conceituais.

Resultados:

- Especificação de Arte do Jogo.

Macro-Iteração 3: Construção da Arquitetura

Duração: 15% do tempo de projeto.

Atividades:

- Construção da arquitetura básica do jogo:
  Frameworks.
  Engine do Jogo.
  Ferramentas.

OBS: Os elementos desta arquitetura básica podem ser construídos ou reutilizados (e adaptados) de outros projetos (ou de terceiros).

Papéis Envolvidos:

- Equipe de Desenvolvimento de Software.

Resultados:
- Arquitetura inicial executável do jogo.
- *Engine* do jogo.
- Ferramentas para a produção do jogo (ex.: Editor de Níveis).

Marco Principal: Arquitetura do Sistema (*Lifecycle Architecture Milestone – LCO)*
- Arquitetura deve estar bem definida e funcional.
- Deve possuir uma visão clara do jogo que será construído (bom planejamento).

5.4.4.3 Produção

A etapa de produção é a mais longa do processo de desenvolvimento de um jogo. O objetivo é construir o jogo de acordo com a visão e plano estabelecidos anteriormente. À medida que o jogo é construído, pequenas mudanças no conceito to jogo podem ocorrer. Contudo, na maioria dos casos, grandes mudanças não são permitidas nesta fase a fim de cumprir o plano dentro do cronograma e custo definidos.

Nesta fase, são construídos protótipos (lançamentos) do jogo. Em jogos menores, espera-se que ao menos dois sejam produzidos: protótipo Alpha e Beta. Em grandes projetos, o número de lançamentos pode ser maior. O protótipo Alpha contém os principais elementos do *gameplay*, o *core gameplay*. O Beta é o jogo completo, com ajustes para serem feitos. As atividades da disciplina de testes são já iniciadas fortemente nesta fase.

No MGUP, a Construção compreende as fases de Produção do Jogo e Produção de Níveis do Game Process genérico (Diagrama 4), que correspondem às duas macro-iterações existentes desta fase. Obviamente, iterações menores podem ser realizadas.

Resumo da Fase de Produção

Macro-Iteração 1: Produção do Jogo

Duração: 40% do tempo de projeto.

Atividades:
- Produção do jogo (software, arte e mídia) de acordo com o planejamento anterior.
- Execução de testes funcionais nas rotinas e módulos do software.

- Execução de testes de jogabilidade realizados pelo próprio pessoal da equipe. Estes são testes minuciosos sobre o *gameplay* e podem ser feitas mudanças consideráveis na mecânica do jogo.
- Divisão da produção em vários lançamentos (mini-projetos) internos e externos.
- Especificação do jogo é sempre verificada de atualizada de acordo com novas questões.
- Execução da etapa de Produção do processo de Criação de jogos (Seção 3.3.2).
- Atualização das ferramentas construídas na etapa anterior de acordo com novas necessidades.
- Produção de mídia é iniciada (principal: som).

Papéis Envolvidos:
- Equipe de Desenvolvimento de Software.
- Líder de Desenvolvimento é responsável por guiar e controlar toda a produção.
- Líder de Som
- Músicos
- Criadores de efeitos sonoros.

Resultados:
- Software do jogo (quase finalizado, pois alguns detalhes são deixados para acertos durante a etapa de construção de níveis).
- Em iterações internas à fase, pode haver lançamentos de versões incompletas do jogo.

Macro-Iteração 2: Produção de Níveis

Duração: 15% do projeto.

Atividades:
- Construção de todos os cenários do jogo.
- Resolução das pendências para a realização desta atividade (elementos do jogo que tenham faltado, partes do software ou mesmo das ferramentas).
- Especificação do jogo é sempre verificada e atualizada de acordo com novas questões.
- Execução da etapa de Produção do processo de Criação de jogos (Seção 3.3.2)..
- Produção do restante da parte de mídia (principal: som).
- Produção dos manuais e materiais promocionais do jogo.

Papéis Envolvidos:
- Equipe de Desenvolvimento de Software.
- Líder de Desenvolvimento.
- Líder de Criação de Jogos.
- Criador de Níveis.
- Parte da equipe de arte (caso seja necessário).
- Líder de Arte.
- Líder de Som
- Músicos
- Criadores de efeitos sonoros.

Resultados:

- Jogo em sua forma quase completa, com todos os níveis.
- Em iterações internas à fase, pode haver lançamentos com versões incompletas do jogo.

Marco Principal: Capacidade Operacional Inicial *(Initial Operational Capability Milestone – IOC)*

- Disponível a primeira versão operacional completa do jogo.

5.4.4.4 Pós-Produção

Nesta fase, existe grande número de esforços voltados para as atividades de teste e validação do jogo. Retornos dos jogadores devem estar basicamente em pequenos ajustes e detalhes de usabilidade. Todas as questões estruturais devem ter sido trabalhadas em fases e iterações anteriores do ciclo de vida.

Resumo da Fase de Pós-Produção

Duração: 15% do tempo de projeto.

Atividades:

- Execução da etapa de Qualidade do processo de Criação de jogos (Seção 3.3.2).
- Realização de testes, validações, pequenos ajustes e modificações, polimentos finais e correções de *bugs*.
- Realização de *Alpha Testing*: Testes com pessoal selecionado – avaliadores especialistas. Mudanças simples são possíveis
- Realização de *Beta Testing*: Testes com usuários finais – jogadores que não possuem conhecimento prévio do jogo. Podem acusar pequenos desvios de usabilidade e interface.

OBS: Muito embora a etapa de Pós-Produção seja fortemente baseada em testes, estes são realizados ao longo de todo o ciclo pelo Criador de Jogos para validações e melhorias no conceito (Ciclo Iterativo de Criação de jogos – Seção 3.3.1).

Papéis Envolvidos:

- Líder de Desenvolvimento de Software.
- Desenvolvedores.
- Líder de Criação de Jogos.
- Criadores de Jogos.
- Líder de Arte.
- Artistas 2D e/ou 3D.
- Líder de Som
- Músicos
- Criadores de efeitos sonoros.

Resultados:

- Entrega final.

Marco Principal: Lançamento do Produto *(Product Release Milestone – PR)*

- Deve estar disponível e entregue a versão final do jogo.

5.4.5 Artefatos e Atividades

Os artefatos estão ligados aos papéis, que executam as atividades. Por sua vez, as atividades geram os artefatos. As atividades não são explicitamente listadas nesta descrição. Ao invés, assume-se que cada artefato tenha inerente consigo uma atividade de produção (ex.: Produzir conceito inicial do jogo → Especificação Conceitual do Jogo).

Como já apresentado, em um processo de jogos existem papéis advindos de diferentes áreas. No entanto, os artefatos produzidos pelos desenvolvedores de software são independentes de domínio. Eles são os mesmos já constantes na definição do RUP: diagramas de classe, colaboração e seqüência, diagramas de componentes, pacotes, arquitetura, e quais outros se façam necessários para melhor planejar e construir o software. O mesmo pode ser dito para a área de gerência de projetos (produtor). Como os artefatos produzidos por estas categorias de papéis são bem conhecidos, não cabe discutir ou apresentar modelos (*templates*) neste trabalho. O objetivo é discutir as peculiaridades de artefatos para o desenvolvimento de jogos. Desta forma, os artefatos a serem discutidos são os das disciplinas de Implementação, Criação de jogos e Testes.

De acordo com o tamanho do projeto pode ser necessária um número maior de artefatos. Como processo derivado do RUP, existe compatibilidade de artefatos entre RUP e MGUP. Nesta descrição serão considerados os artefatos específicos para jogos, assim como, aqueles que fazem parte dos Artefatos-Chave do RUP Lite (seção 4.3.1.1).

Os artefatos apresentados são pertencentes a duas categorias: Artefatos de Controle e Artefatos de Produção. Os Artefatos de Controle são documentos e diagramas utilizados para, dentre outras funções, guiar, monitorar ou planejar a criação dos artefatos de Produção. Exemplos: Diagrama de Classes, Plano de Projeto, Glossário, Especificação de Arte, etc. Artefatos de Produção são o resultado direto das atividades de criação de elementos do jogo.

Exemplos: Código Fonte, Ferramenta, Arte 2D, Modelo 3D, Arte Conceitual, etc. Com o intuito de simplificar a explicitação do tipo de artefato, serão usadas as letras C (Controle) e P (Produção).

Artefatos de Produção não possuem modelos. Já Artefatos de Controle possuem exemplos, modelos e especificações que podem ser encontrados nas referências deste trabalho:

- Exemplos e modelos de artefatos relacionais ao RUP: [POLLICE et. al. 2003], [KROLL & KRUCHTEN 2003], [KRUCHTEN 2000] e [LARMAN 2001].
- Exemplos e modelos de artefatos relacionados ao Desenvolvimento de Jogos: [FLYNT 2004], [RUCKER 2002], [GAMASUTRA 2005], [BETHKE 2003], [ROUSE 2000], [FULLERTON 2004] e [GAMEDEV 2005].

Modelos de artefatos de controle e exemplos de artefatos de produção estão disponíveis nos Anexos I e II deste trabalho.

5.4.5.1 Artefatos de Criação de jogos

Os artefatos de criação de jogos são, basicamente, documentos descritivos de todos os elementos do jogo. Em jogos pequenos, geralmente existe uma Especificação Conceitual do Jogo e uma Especificação do Jogo. Já em jogos maiores, a especificação detalhada é subdividida em outros documentos, que tratam de assuntos específicos do conceito do jogo.

- (C) Especificação Conceitual do Jogo: apresenta uma visão geral do jogo por meio de texto e imagens. Geralmente apresenta uma primeira visão do jogo, descrevendo de forma superficial o conjunto de elementos fundamentais do *gameplay*, o *core gameplay*.
- (C) Especificação do jogo: apresenta de forma detalhada todos os elementos da mecânica e concepção de um jogo. Contêm outros artefatos que juntos especificam todos os outros detalhes do jogo, como artes conceituais, *mockups*, *story boards* e

especificações de níveis e de personagens. Desta forma, pode-se dizer que estes outros artefatos são na verdade, sub-artefatos (ou partes) da Especificação do jogo. Em jogos onde uma ou mais destas partes necessitem de um bom detalhamento, elas podem vir a se tornar realmente documentos separados e referenciados pela Especificação do jogo.

- (C) e (P) Especificação de Níveis (*Level Design*): apresenta os cenários do jogo de forma detalhada, na forma de um mapa, geralmente criados em softwares específicos: os chamados editores de níveis (*level editor)*. Para questões de documentação, os mapas podem ser impressos ou mesmo planejados com antecedência em papel.
- (C) Especificação de Mecânicas do Jogo: apresenta os principais elementos do *gameplay*: o *core gameplay*. È constituído de regras do jogo, objetivos, recompensas, forma de interações, dentre outras coisas.
- (C) Especificação de Personagens: apresenta todos os detalhes descritos dos personagens do jogo. Contêm informações sobre o histórico do personagem (*back story*), suas características (armas, habilidades, personalidade, aparência, etc) e também, detalhes de como deve se comportar no jogo (descrição em alto nível da forma como se espera que o personagem aja no jogo).
- (C) Especificação da Física e I.A. do Jogo: contém detalhes em alto nível da forma como se espera que a física e a mecânica de inteligência artificial do jogo funcione. Imagens e diagramas informais são geralmente usados para ilustrar as explicações textuais deste documento.
- (C) Especificação do Roteiro do Jogo (Script): apresenta toda a estória e seqüência de eventos que ocorrem no decorrer do jogo. Neste sentido, detalha quais personagens e eventos, assim como quaisquer outros elementos, estão presentes em cada momento da estória do jogo.
- (C) Especificação de Padrões de Navegação, Telas e Fluxo: apresenta uma

descrição de como se espera que seja a forma de navegação e interação nas telas, menus, elementos de interface gráfica (botões, caixas de texto, etc.) e painéis do jogo. Também possui um diagrama informal que demonstra o fluxo de telas que o jogo deve seguir e os eventos que definem os caminhos específicos. Também apresenta uma descrição de quais informações (e o estilo) que cada tela do jogo deve possuir.

5.4.5.2 Artefatos de Implementação

5.4.5.2.1 Desenvolvimento de Software

- (C) Especificação Técnica: Apresenta todo o planejamento de software feito para a produção do jogo. Ele é preparado durante a fase de Pré-Produção. Os principais tópicos abordados por este documento são:
  - (C) Arquitetura: Descrição textual e com o uso de diagramas (classe, pacotes e/ou componentes) da arquitetura planejada para o jogo. Esta parte é tipicamente modificada ao longo do projeto, pois a arquitetura está em constante evolução durante a produção do software.
  - (C) Soluções: Descrição textual e com o uso de diagramas (classe, pacotes, componentes, colaboração, seqüência, fluxogramas, notações informais, etc) que demonstram o planejamento inicial das soluções mais complexas do software do jogo (em termos de algoritmos, estruturas de dados e arquitetura).
  - (C) Problemas e Riscos: Listagem dos problemas e riscos que podem vir a atrasar o projeto. Eles devem ser verificados o mais cedo possível.
- (C) Documento de Requisitos: Caso o jogo seja complexo, talvez se faça necessário a criação de um documento de requisitos, contendo os requisitos funcionais e não-funcionais.
- (P) Código: O principal artefato produzido por desenvolvedores de software é o

código fonte, e, conseqüentemente, o executável gerado a partir deste. Existem diferentes categorias de código gerado pelas diferentes especialidades (papéis) de desenvolvedores tipicamente encontrados em uma equipe de jogos: estrutura e lógica do jogo, *frameworks*, *engines*, componentes e ferramentas.

5.4.5.2.2 Artes Gráficas

A maior parte dos artefatos de arte gerados está relacionada à produção do jogo. O único artefato de planejamento é a Especificação de Arte.

- (C) Especificação de Arte: Contém todo o planejamento artístico (visual) para o jogo – desenhos conceituais (esboços de criações), *mockups* de interface (projeto de telas e interface), etc. – e definições – unidade de medida no espaço 3D, tamanhos dos objeto, etc.
- (P) Arte Conceitual: Desenho feito à mão (em preto e branco ou colorido) para ilustrar cada elemento do jogo. O conjunto formado por todos os desenhos serve de base para a produção da arte real do jogo, tanto 2D quanto 3D.
- (P) Arte 2D: Representa a arte final 2D do jogo. Tem como ponto de partida a Arte Conceitual. A arte 2D inclui todos os elementos visuais 2D, incluindo personagens, animações, cenários e elementos de interface (botões, fontes, barras indicativas, etc.).
- (P) Modelos 3D: Modelos tridimensionais (estruturas de arame ou malha) de elementos do jogo, incluindo personagens e objetos, cenários. Este trabalho é baseado na Arte Conceitual. Possui relação direta com dois outros artefatos: Animações 3D, que dão movimento aos modelos e, Texturas, que acrescentam o visual ao modelo (o arame mostra a forma).
- (P) Personagens 3D: Especificação de Modelos 3D. Os jogos geralmente possuem vários personagens. Este artefato possui dependência da Arte Conceitual e relacionamento com Texturas e Animações 3D.

- (P) Animações 3D: As animações 3D são arquivos que aplicam movimento sobre modelos 3D. São especialmente usados em personagens e possuem relação direta com Personagens 3D (e Modelos 3D).
- (P) Texturas: Imagens que são utilizadas para dar o visual de uma malha 3D. Possuem relação direta com Modelos 3D e Personagens 3D.
- (P) Story Board: São seqüências de cenas similares a uma estória em quadrinhos. São usados principalmente para descrever visualmente uma seqüência de acontecimentos no jogo ou uma *cutscene* (animação que conta parte da estória).

5.4.5.2.3 Mídia Sonora

- (C) Especificação de Mídia Sonora: Define quais as exigências de músicas e sons para o jogo.
- (P) Música: Arquivos de músicas compostas em software profissional específico. Também pode referenciar as partituras.
- (P) Efeitos Sonoros: Arquivos de efeitos sonoros compostos em software profissional específico.

5.4.5.3 Artefatos de Testes

Os artefatos de testes mais comuns são os Planos de Testes Funcionais e Não-Funcionais. Estes, por já serem adotados no RUP são assimilados naturalmente pelo MGUP. Para o desenvolvimento de jogos para celular, entram três artefatos mais específicos. Os resultados dos testes podem ser anotados em documentos informais, tais como planilhas.

- (C) Plano de Testes de Jogabilidade: A qualidade em software é medida por vários atributos: atendimento às funcionalidades requisitadas como, por exemplo, robustez, confiabilidade, disponibilidade e desempenho. Em jogos, quesitos novos são acrescentados e os principais são: diversão, imersão e captação. O teste de jogabilidade serve para medir esses atributos.

- (C) Plano de Testes de Usabilidade: Pode ser considerado uma especialização do teste de jogabilidade direcionado à análise da compreensão do jogador sobre informações e interfaces do jogo.
- (C) Plano de Testes de Compatibilidade: O desenvolvimento seguro de aplicações para celulares exige que normas e padrões sejam seguidos. Um exemplo é o TBT – *True Brew Testing* ([BREW 2005]) – que especifica uma série de requisitos não-funcionais com os quais a aplicação Brew precisa estar em concordância. Cada fabricante tem suas exigência, mas existe um consenso comum em torno do que é necessário para uma aplicação.

5.4.6 Papéis

Os papéis envolvidos na produção de jogo no modelo MGUP são organizados em agrupamentos lógicos e baseados em [PEDERSEN 2003], [BETHKE 2003], [ROUSE 2000] e [FULLERTON 2004].

5.4.6.1 Planejamento e Produção

Os papéis de planejamento e produção de uma equipe de jogos são geralmente desempenhados pelas pessoas mais experientes do time. Juntamente com o Líder de Criação de Jogos, eles são os responsáveis por tomar as decisões mais cruciais. Em projetos menores, papéis como Planejador de Software, Arquiteto, Programador, Planejador de Testes e Testador são exercidos pelas mesmas pessoas. Neste caso, algumas empresas usam uma nomenclatura diferenciada para tal papel que representa um agrupamento de vários outros: Engenheiro de Software, Analista Desenvolvedor ou Engenheiro de Sistemas.

- Produtor: Um produtor é um gerente de projeto. Ele é responsável por organizar de forma eficiente o cronograma de trabalho para as tarefas definidas pelo Planejador e Arquiteto de Software. Ele também lida com as interações entre os membros do time e age como uma interface entre o time do projeto e o departamento de gerenciamento e

marketing da organização (ou o time de publicação). Também realiza interface com os interessados no projeto a fim de garantir que seus anseios estejam sendo levados em conta no desenvolvimento do jogo. Artefatos produzidos: Plano de Projeto, Plano de Desenvolvimento e Glossário.

- Líder de Criação de Jogos: O papel do líder de criação de jogos é realizar a concepção geral do jogo e seus elementos-chave (mecanismos, diversão do jogo, estória e ambientação, etc). A partir desta, ele a divide em partes e as designa para os outros Criador de Jogoss. Assim, detalhes de cenários, personagens e mecânicas são definidos por outros papéis de criação de jogos. O Líder de Criação de Jogos deve estar em constante comunicação com toda a equipe para garantir que todos estejam compartilhando uma visão correta do jogo, além de investigar viabilidades e garantir que todos tenham o que for necessário para continuar com a produção do jogo. Toda a parte de concepção do jogo está sob a responsabilidade do Criador de Jogos, que pode ser desempenhado por mais de uma pessoa. Artefato produzido: Especificação do jogo.
- Líder de Música: Responsável por coordenar as atividades dos demais integrantes do grupo de mídia sonora. Também conhecido como Diretor de Som. A presença de um líder nesta área é imprescindível quando o jogo possui grande investimento nesta parte e tem-se uma equipe numerosa (ou quando existe sub-contratação de serviços). Artefatos produzidos: Especificação de Mídia Sonora e qualquer outro artefato de mídia sonora.
- Líder de Desenvolvimento: Também conhecido como Diretor Técnico. Coordena e monitora o trabalho do time de desenvolvimento para assegurar que o cronograma está sendo seguido. Também realiza interface com o produtor para assegurar que o cronograma está sendo seguido e reporta progressos para que este possa ser mensurado, assim como eventuais problemas possam ser resolvidos o mais cedo

possível. O desenvolvedor líder é o mais bem preparado tecnicamente e é responsável pela integridade do software sendo desenvolvido. Cerca de três quartos de seu tempo é gasto com desenvolvimento, enquanto o restante é usado para realizar tarefas administrativas. Programadores líderes são geralmente selecionados baseando-se em critérios técnicos e não em qualquer outro critério. Artefatos produzidos: Especificação Técnica e qualquer outro artefato de desenvolvimento.

- Líder de Arte: Também conhecido como Diretor de Arte, o Artista Líder é a principal interface do grupo de arte para com o restante da equipe, em especial a parte gerencial. Também é responsável por acompanhar o trabalho de todos os demais papéis de arte assim como ajudar em tarefas de arte. Entende-se que o líder é a pessoa mais bem preparada da equipe de arte gráfica. Por fim, ele deve servir de apoio ao grupo de criação de jogos na definição de toda a parte visual do jogo. Artefatos produzidos: Especificação de Arte e qualquer artefato de produção.
- Líder de Qualidade: O papel do líder de qualidade é supervisionar o time de qualidade (QA) e cooperar com o produtor e o Criador de Jogos a fim de assegurar-se de que o jogo foi consistentemente testado, tanto pelo lado do *gameplay* quanto pelos requisitos funcionais e não-funcionais. O líder de qualidade tem como atividades projetar o plano de testes e atribuir tarefas de testes aos diferentes integrantes do time de QA. Os resultados dos testes serão reportados ao produtor.
- Planejador de Software: O planejador de software traduz a especificação do jogo (Especificação do Jogo) em um conjunto de requisitos técnicos detalhados. Geralmente, ele trabalha em conjunto com o Criador de Jogos e o arquiteto para preparar as especificações técnicas detalhadas. Junto ao arquiteto, o planejador é responsável por definir as tarefas a serem implementadas no software, estimando tempo e esforço necessários. O papel de planejador é temporário, não se fazendo necessário ao longo de todo o projeto. Suas atividades estão concentradas nas

primeiras fases do projeto. Artefato produzido: Documento de Requisitos.

- Arquiteto de Software: Traduz as especificações de requisitos em um documento técnico detalhado (arquitetura). Ele é responsável por toda a arquitetura do software, assim como questões técnicas e problemas de implementação que envolvem a definição da arquitetura. Teoricamente, o projetista é o desenvolvedor mais experiente (possui mais conhecimento) sobre o software de jogos. Artefato produzido: Especificação Técnica.

5.4.6.2 Desenvolvimento de Software

A estrutura dos papéis envolvidos no desenvolvimento do software de um jogo está presente, praticamente, da mesma forma como em outros domínios de aplicações. A diferença maior está no tipo de conhecimento que estes profissionais possuem.

Um Sistema de Informação (SI) é projetado levando-se em conta acomodar todas as regras do negócio da empresa interessada, modularização (visto que esses sistemas são geralmente grandes), independência de plataforma e tecnologia. A modelagem das regras de negócio é geralmente seqüencial e não exige resultados na ordem de milisegundos. Programadores possuem um foco no domínio das diversas tecnologias que são integradas na solução de forma a acelerar e padronizar a produção, e que permitem ao sistema comunicar-se com outros em funcionamento. O foco de um SI está no controle de fluxo e no volume das informações necessárias ao negócio do interessado no software. Aqui os interessados são companhias e o software é visto como uma solução. Muitas vezes a solução não é inovadora, no sentido de se estar criando algo diferente. Softwares de SI podem ser descritos como um mecanismo de automatização, facilitação e aceleração do funcionamento de um setor ou empresa. Os engenheiros têm a responsabilidade de automatizar e encontrar pontos de otimização no fluxo de funcionamento de uma organização. No final, as regras definidas são projetadas e implementadas em software.

Por outro lado, no desenvolvimento de jogos, os profissionais precisam estar preocupados com outros elementos, como: diversão do usuário, atendimento ao público alvo, desempenho da aplicação, recursos computacionais disponíveis, algoritmos e estruturas de dados, matemática, simulações físicas, inteligência artificial e computação musical, além de outros. Um jogo é geralmente encarado como um produto que deverá ser adquirido por uma grande quantidade de usuários finais. Mas existem casos onde os jogos são considerados uma prestação de serviços: em especial, os Jogos Multiplayer Massivos. Nesses jogos, os usuários adquirem o título e pagam uma mensalidade para o uso dos servidores. Os jogadores interagem uns com os outros por meio do jogo e a comunicação dos computadores é gerenciada por servidores dedicados. O produtor do jogo realiza um trabalho constante de manutenção do software e ajustes no jogo de forma a melhorar a experiência interativa do usuário.

A configuração de papéis apresentada para a área (ou grupo) de desenvolvimento de software segue uma linha genérica. Em determinados projetos, outros papéis podem vir a ser necessários. No caso de jogos que utilizam servidores Web, o papel de Administrador de Banco de Dados pode ser necessário.

Na parte de desenvolvimento, encontram-se os papéis responsáveis pela implementação das tarefas e arquitetura definidas pelo projetista, de forma a satisfazer os requisitos elicitados pelo analista. Uma pequena estrutura é definida para os desenvolvedores, com um líder (responsável por ajudar nas tarefas e garantir a definição da arquitetura) e um time de desenvolvedores especializados em áreas diferentes.

- Desenvolvedor: Este papel representa a base do processo de desenvolvimento e é o responsável direto pelo progresso (ou não) do projeto. Seu trabalho envolve implementar o que é especificado e definido pelo analista e pelo arquiteto. Um desenvolvedor é geralmente responsável por uma determinada parte (módulo ou componente) do software. O desenvolvedor é pelo software, incluindo os

experimentos iniciais (testes, provas de conceito e protótipos), testes unitários, testes de integração e correção de defeitos (*bugs*). É um trabalho que envolve tanto pesquisa para novas soluções, quanto desenvolvimento. De uma forma geral, pode-se dizer que três grupos de desenvolvedores estão envolvidos no desenvolvimento de um jogo: desenvolvedores de ferramentas, da *engine* do jogo ou da sua lógica. Artefatos produzidos: Especificação Técnica e qualquer outro artefato de desenvolvimento.

- Desenvolvedores de Ferramentas: Seu papel é desenvolver softwares, plugins ou módulos do sistema principal que auxiliem os demais integrantes da equipe a desempenharem seus papéis. Exemplos de ferramenta são: um compilador específico, um software de compactação ou gerenciador de recursos do jogo, um editor de mapas, um software para criação das animações dos personagens do jogo ou um *plugin* de conversão de um formato de arquivo de outras ferramentas (ex.: Photoshop, 3D Studio Max, Maya, Sound Forge, etc) para que possa ser utilizado na *engine* do jogo. Artefatos produzidos: Código (e sua Documentação).
- Desenvolvedor da *engine* do Jogo: A *engine* do jogo é a parte principal de seu software. Pode ser desenvolvido na forma de um *middleware* ou como um conjunto de componentes. A *engine* é responsável por gerenciar recursos (sons, músicas, imagens, modelos, animações, textos, etc), prover elementos fundamentais do jogo (mecanismos de física, de inteligência artificial, de interação com cenários e objetos, etc) e abstrair a complexidade destes elementos para o desenvolvedor da lógica do jogo. Uma *engine* funciona como uma camada de abstração dos recursos fundamentais de um jogo. Artefatos produzidos: Código (e sua Documentação).
- Desenvolvedor da Lógica do Jogo: O papel deste desenvolvedor está diretamente ligado às características do *gameplay* do jogo. Ele é responsável por implementar todo o conteúdo do jogo, sobre a *engine*: regras, objetivos, personagens,

animações, cenários, músicas e efeitos sonoros, objetos, regras de interações e I.A. (Inteligência Artificial aplicada à Jogos), telas, etc. Artefatos produzidos: Código (e sua Documentação).

5.4.6.3 Garantia de Qualidade

O grupo de qualidade e suporte inclui o time de testes, que assegura que o jogo está com qualidade e jogabilidade consistente. O processo de testes de jogos é tanto qualitativo quanto quantitativo. É qualitativo no sentido de calibrar o *gameplay* e quantitativo no sentido de encontrar *bugs*.

- Técnico de Qualidade: O papel do técnico de qualidade é testar o código escrito pelos desenvolvedores. O teste deve cobrir todos os caminhos possíveis de execução do código, não importando o quão simples ele pode ser. Para tanto, ele deve interagir com o desenvolvedor para garantir que o plano de testes cobre todos os caminhos possíveis. De fato, este é um trabalho bastante minucioso e é considerado o teste mais detalhado. Portanto, este papel pode ser também de um desenvolvedor. O tipo de teste executado é o chamado "Teste Caixa Branca", pois a forma de funcionamento interna do que está sendo testado é conhecido. Ao contrário, no "Teste Caixa Preta" testa-se os resultados da implementação. Este é também chamado de Teste Funcional e pode ser executado por qualquer um que tenha o plano de testes em mãos. Artefatos produzidos: Planos de testes em geral.
- Testador de Jogabilidade: Este papel é responsável por jogar o jogo e fornecer retornos à equipe de produção. Inicialmente, os "testadores jogadores" são os desenvolvedores e artistas do projeto. A partir do meio até o final do projeto a importância destes testes aumenta e passa a existir a necessidade de se fazê-los de forma precisa. Existem cinco opções de perfis de testadores para se montar uma equipe de testes de jogabilidade. Geralmente, a equipe é composta de diferentes perfis.

Esses perfis são apresentados na Tabela abaixo. Alguns fatores que influenciam na definição da equipe de testes de jogabilidade são o tamanho da organização e orçamento do projeto. Usa o Plano de Testes de Jogabilidade e o Plano de Testes de Usabilidade.

Perfil de Testador	Descrição	Atuação nos Testes
Integrante da Equipe	Integrante da equipe do projeto ou de outros projetos de jogos.	Os integrantes da equipe são ideais para a realização de testes que possam acarretar em mudanças técnicas e de jogabilidade severas. São usuários bastante meticulosos e observam detalhes técnicos e falhas de implementação e no conceito do jogo. Dependendo do objetivo do teste, podem não ser uma boa opção porque podem estar muito familiarizados com o jogo.
Testador Eventual	Jogador que é convidado para fazer alguns testes no jogo por sua experiência.	Estes testadores realizam testes com o objetivo de verificar a inteligibilidade e aceitabilidade do *gameplay*. Geralmente, empresas pagam alguns jogadores indicados por integrantes da equipe para testarem os jogos. Estes testes
Testador Permanente	Jogador contratado como testador permanente da empresa, possivelmente fazendo parte de uma equipe de testes de jogabilidade.	Companhias maiores possuem um departamento específico para testes de jogabilidade e contratam pessoas para este fim. Neste caso, os testadores podem fazer uma análise completa do jogo, indo à procura de *bugs* até a análise da inteligibilidade e aceitabilidade do *gameplay*. O perfil dos integrantes pode variar entre assíduos, freqüentes ou casuais. O custo desta equipe pode ser compensado à medida que seus integrantes participem de vários projetos ao mesmo tempo.
Testador Terceirizado	Jogador ou empresa especializada de testes de jogabilidade que é contratado para atuar em um projeto.	Caso não haja fluxo suficiente para se manter uma equipe interna de testes, este serviço pode ser terceirizado com uma "agência de jogadores teste". A vantagem do uso destas agências é que podem fornecer testadores com os perfis necessários ao teste.
Testador Beta Público	Consumidor final do jogo.	Uma alternativa usada por muitas empresas é o teste beta direto com o público-alvo do jogo. Neste cenário, uma versão quase finalizada do software (*release beta*) é disponibilizada ao público em geral. Também, é comum que o jogo seja limitado de alguma forma, não contendo todas as funcionalidades da versão final, pois caso contrário, a fabricante estaria disponibilizando a versão final do jogo de forma gratuita a seus potenciais consumidores. A distribuição da versão beta pode ser liberada à um público seleto ou não (por revistas, CD-ROMS e web).

Tabela 3: Perfis de Testadores de Jogabilidade.

5.4.6.4 Criação de jogos

Os papéis encontrados nesta parte de uma equipe de desenvolvimento de jogos são os responsáveis por toda a concepção do jogo, incluindo história, ambiente, cenários, personagens, mecânicas, regras e todos os elementos. Dependendo da complexidade do projeto, o papel de Criador de Jogos (Criador de Jogos) pode ser realizado por uma pessoa ou por um time. Caso seja necessário um time, cada integrante assume a responsabilidade de determinadas partes da concepção do jogo.

No desenvolvimento de jogos, o papel do Analista/Engenheiro de Negócios é representado pelo Criador de Jogos, mas há uma grande diferença entre eles: no primeiro caso, o negócio é externo (pertence ao cliente) e o analista o investiga e documenta; já no segundo, o negócio é proposto e concebido pelo próprio Criador de Jogos (o cliente apenas valida seu ponto de vista sobre o conceito do jogo).

O papel de um Criador de Jogos é conceber os jogos que o time desenvolve. Ele produz um documento inicial de criação de jogos, que contém os elementos-chave do jogo (um documento de visão do jogo) e depois, o documento de Especificação do jogo, que contém todas as informações necessárias ao desenvolvimento do jogo, cobrindo cada elemento de forma abrangente e detalhada. O Criador de Jogos trabalha em conjunto com o analista e o projetista a fim de averiguar a viabilidade de determinados elementos do jogo.

De uma forma geral, é de responsabilidade de um Criador de Jogos não só produzir os artefatos necessários ao desenvolvimento do jogo mas também deve: trabalhar junto ao pessoal de arte para definição das imagens conceituais do jogo; averiguar com o pessoal de Mídia Sonora as possibilidades em termos de músicas e efeitos sonoros, e acompanhar a evolução do desenvolvimento do jogo, garantindo que seu conceito esteja sendo implementado de forma correta. O Criador de Jogos também deve realizar as atualizações necessárias em toda a documentação do jogo à medida que mudanças no conceito do jogo

foram sendo realizadas. Essas mudanças constituem um dos elementos-chave para a correta definição do jogo, pois é comum que alguns elementos não funcionem tão bem na prática (no jogo em funcionamento) quanto foram concebidos.

Os papéis que envolvem a concepção de um jogo, de forma geral, estão apresentados abaixo. Em projetos e organizações menores, estes papéis podem todos ser atribuídos a um único Criador de Jogos ou então, alguns podem ser passados a outros integrantes da equipe que tenham capacidade para sua realização, como é o caso do papel de criador de cenários (*level designer*).

- Criador de Jogos: O papel do Criador de Jogos é definir elementos do jogo. Representa o papel macro da criação de jogos. Todos os outros papéis de criação de jogos são especializações deste. Artefatos produzidos: pode manter qualquer um dos artefatos de criação de jogos.
- Criador de Níveis: O *level designer* é o Criador de Jogos responsável por realizar a concepção dos cenários e ambientes do jogo. Não necessariamente ele é responsável por fazer a criação dos elementos gráficos dos ambientes (isso pode ser delegado aos artistas gráficos). Ele pode trabalhar em cima de uma biblioteca de peças criadas por artistas e definir toda a estrutura dos cenários. Essa definição também pode incluir o posicionamento dos personagens, objetivos e eventos-chave do ambiente. Artefato produzido: Especificação de Níveis
- Criador de Personagens: O *character designer* é o Criador de Jogos responsável por conceber todos os personagens do jogo. Ele deve trabalhar em conjunto com um artista gráfico para definir a parte visual dos personagens. No entanto, um personagem não é somente constituído de elementos visuais: há sua personalidade, histórico, características frente aos mecanismos do jogo e comportamentos. Este papel pode ser atribuído diretamente a um artista gráfico, caso os personagens sejam bastante simples e necessitem basicamente de seus elementos visuais. Artefato produzido:

Especificação de Personagens.

- Criador de Mecânicas: É o Criador de Jogos responsável por definir os principais elementos do *gameplay* (o *core gameplay*). Artefatos produzidos: Especificação de Mecânicas do Jogo, Especificação da Física e I.A. do Jogo e Especificação de Padrões de Navegação, Telas e Fluxo
- Criador de Estória e Roteiro: O *story and script designer* é um tipo de Criador de Jogos que fica responsável por criar toda a estória e enredo do jogo, como eventos, explicações, tramas, entre outros. Este é um papel bastante dependente do tamanho e do tipo do jogo a ser desenvolvido no projeto. Em jogos típicos de ação, por exemplo, a estória tem um papel bastante superficial. Artefato produzido: Especificação do Roteiro do Jogo.

5.4.6.5 Artes Gráficas

A parte de arte gráfica de um projeto de jogos consiste de um conjunto de artistas gráficos com diferentes habilidades (especialidades). O trabalho de um artista gráfico é, basicamente, elaborar e produzir toda a parte visual de um jogo, podendo estar agindo em conjunto com um Criador de Jogos para definição de características gráficas que sejam determinantes para a concepção do jogo.

5.4.6.5.1 Arte 2D

Entende-se por arte bidimensional (2D) todos os desenhos, *layouts*, fotos, animações e efeitos especiais com base em desenhos. Os artistas 2D trabalham principalmente com ferramentas de desenho e edição de imagens. Em jogos 2D, eles são responsáveis por toda a produção visual. Em jogos 3D, eles geralmente são responsáveis pela concepção inicial dos personagens, pela "pele" (texturas) dos modelos e por toda diagramação de telas e menus do jogo.

- Artista Conceitual: O papel do artista conceitual é trabalhar junto ao Criador de Jogos

na criação do conceito dos elementos gráficos do jogo: personagens, cenários, objetos, itens, telas, storyboards, etc. Seu trabalho pode ser encarado como um guia para a produção dos elementos visuais do jogo, embora não seja necessário que todos eles tenham uma versão inicial conceitual. Artefato produzido: Arte Conceitual.

- Criador de Story Board: É uma especialização de um Artista Conceitual, responsável por criar imagens conceituais de cenas animadas. Artefato produzido: Story Board.
- Artista 2D: A responsabilidade deste papel é criar a arte final do jogo. Quando disponível, ele tem como ponto de partida o trabalho do artista conceitual. Suas criações incluem todos os elementos visuais 2D de um jogo, incluindo objetos estáticos, animações, cenários e elementos de interface (botões, fontes, barras indicativas, etc.). Artefato produzido: Arte 2D.
- Criadores de Texturas: Seu papel é criar a arte 2D das texturas dos modelos 3D de um jogo tridimensional. As texturas podem ser consideradas um invólucro que vai sobre o modelo, que possui a forma e animação. Artefato produzido: Texturas.

5.4.6.5.2 Arte 3D

A arte 3D compreende basicamente a parte de modelagem e animação de objetos 3D. Animação pode ser tanto de personagens como tomadas de cenas, usando movimentos de câmera. Os artistas 3D também criam efeitos especiais para as cenas tridimensionais. Em alguns casos, são também responsáveis por criar vídeos que completam a trama do jogo. Esses vídeos são geralmente produzidos por equipes separadas de animação e contam com a participação de artistas 2D para a criação das texturas e arte conceitual.

- Modelador 3D: Seu papel é criar os modelos tridimensionais que são usados no jogo. Esses modelos podem ser usados para qualquer elemento, incluindo personagens, cenários, itens, obstáculos e elementos de interface (GUI). O modelador 3D pode ser dividido em vários papéis, de acordo com o tipo de objeto com o qual irá trabalhar.

Artefato produzido: Modelos 3D.

- Modelador de Personagens: Representa uma especialização de Modelador 3D. A necessidade de um papel como este vem da existência de grande quantidade de personagens em vários tipos de jogos. Artefato produzido: Personagens 3D.
- Animador 3D: Seu papel é criar as animações para os modelos trabalhados pelo Modelador 3D. Artefato produzido: Animações 3D.
- Iluminador 3D: O iluminador 3D é responsável por definir toda a iluminação que é aplicada aos objetos 3D. Artefato produzido: Modelos 3D.

5.4.6.5.3 Mídia Sonora

O elemento considerado de maior importância para o envolvimento do jogador com o clima do jogo é o som. O trabalho que é realizado com a parte sonora em um jogo é tão importante quanto a realizada na produção de um filme. A Mídia Sonora de um jogo é composta por músicas e os efeitos sonoros.

- Músico: O papel do músico é criar todas as músicas de um jogo. Geralmente, o músico trabalha de forma separada do resto do time. Geralmente, seu trabalho se inicia quando o jogo já está em um estágio intermediário de desenvolvimento. Ele recebe uma especificação ou demonstrativo para que possa estabelecer o estilo musical de acordo com a ambientação do jogo. Atualmente, tem-se tornado mais comum o uso de música interativa (que sofre modificações de acordo com acontecimentos no jogo), e isso faz com que o músico tenha um papel mais integrado e presente durante uma parte maior do processo de desenvolvimento, e não somente ao final. Artefato produzido: Música.
- Criador de Efeitos Sonoros: É de responsabilidade deste papel, criar todos os efeitos sonoros do jogo, incluindo vozes, efeitos reais capturados (explosões, sons de passos, fluídos, etc) e efeitos sintéticos (raios lasers, efeitos especiais, sons de personagens,

etc). Assim como o músico, o criador de efeitos sonoros é um papel mais separado do resto da equipe. Artefatos produzidos: Efeitos Sonoros.

5.4.6.5.4 Técnicos Específicos

Aqui está sendo apresentado um quinto grupo de papéis envolvidos no desenvolvimento de jogos que agrupa todos aqueles provindos de áreas distintas das outras apresentadas até o momento. A existência de alguns desses profissionais em um projeto de jogos depende do tema e complexidade do jogo. Exemplos desses papéis são: técnicos de captura de movimento, que trabalham na produção das animações; consultores militares, engenheiros mecânicos de carros, motos ou aeronaves, que podem ser necessários caso o objeto do jogo seja buscar o máximo de fidelidade com elementos do mundo real; e, atores e dubladores de personagens.

Esses papéis podem não estar diretamente integrados à equipe de produção do jogo. Eles geralmente são contratados por meio de serviços terceirizados com estúdios especializados.

5.4.7 Disciplinas

As disciplinas do MGUP têm um formato similar ao apresentado no RUP. Todavia, há um conjunto de modificações, conforme apresentado no texto. A maior delas está na disciplina de Criação de jogos, que substitui a de Modelagem de Negócio. Pequenas alterações são feitas nas disciplinas de Implementação e Testes.

5.4.7.1 Disciplina de Criação de jogos

Esta disciplina possui no MGUP um papel semelhante à Modelagem de Negócios no RUP. Ela define todos os detalhes do software a ser produzido. Uma descrição detalhada do propósito desta disciplina está na seção 3.3 (Criação de jogos).

Os papéis, atividades e artefatos contidos nesta disciplina já foram discutidos nas seções 5.4.5 e 5.4.6. Esta seção reúne todos esses elementos e apresenta o *workflow* da

disciplina (Diagrama 5). O comportamento do *workflow* é baseado no processo de criação de jogos (seção 3.3.2), no ciclo iterativo de criação de jogos (seção 3.3.1) e nos papéis e artefatos produzidos, descritos no MGUP.

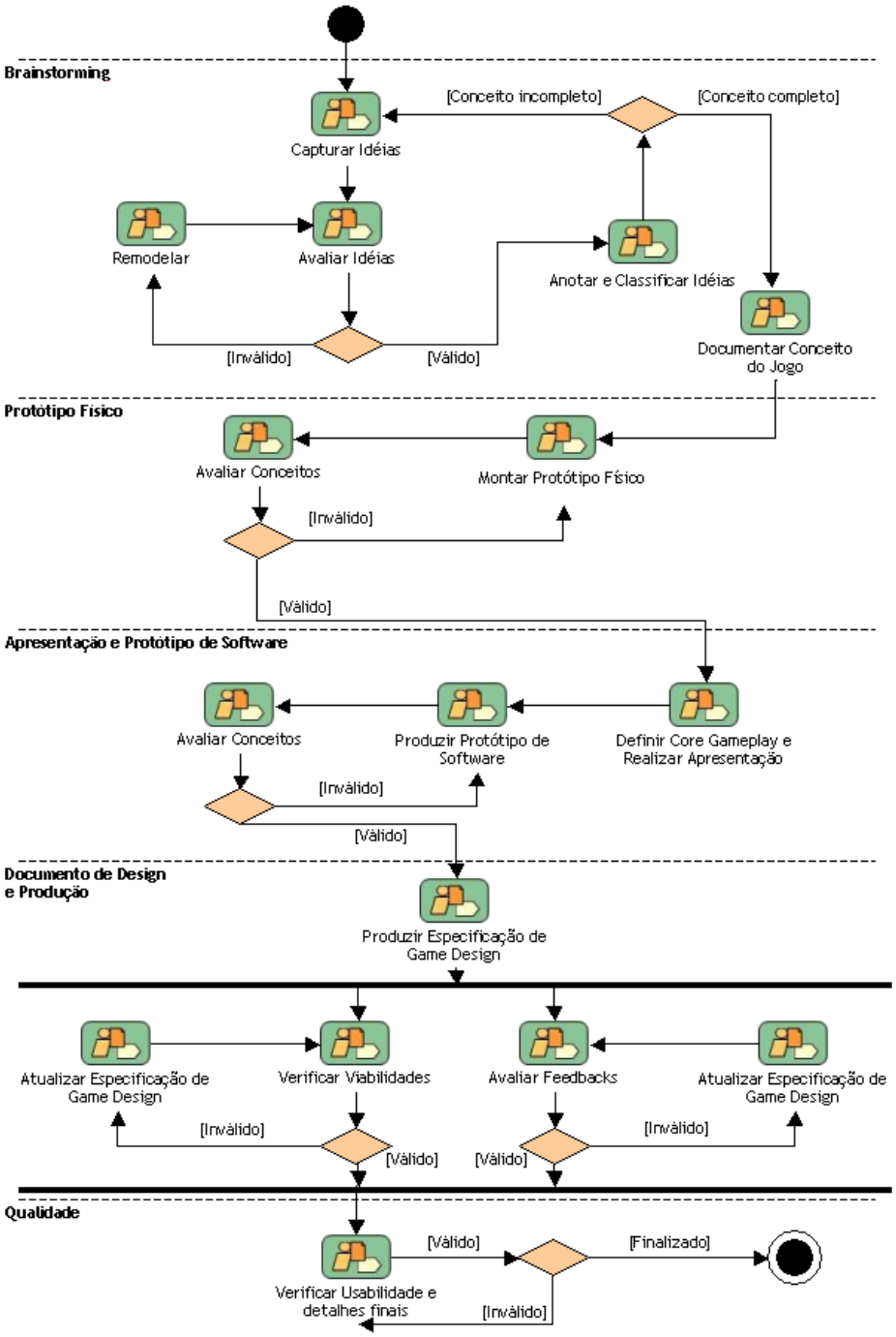

Diagrama 5: Workflow da disciplina de Criação de jogos.

5.4.7.2 Disciplina de Implementação

Implementar um jogo significa construir seu software, sua arte e a mídia sonora. Desta forma, pode-se dizer que a disciplina de implementação do MGUP contempla esses modelos de produção. Pode se considerar que esta disciplina contém três sub-disciplinas: Implementação de Software, Implementação de Arte e Implementação de Som.

Os artefatos, atividades e papéis de desenvolvimento, arte e mídia sonora estão apresentados nas seções 5.4.5 e 5.4.6. Como só existe um artefato de controle para arte e outro para mídia e o restante dos artefatos são de produção, os *workflows* dessas sub-disciplinas são mais simples e bastante semelhantes ao processo de criação de jogos (baseados em construção incremental e retornos). O *workflow* da sub-disciplina de Implementação de Software é o mesmo adotado no RUP (ver [KRUCHTEN 2000]).

5.4.7.3 Disciplina de Teste

A disciplina de Testes do MGUP apresenta peculiaridades adicionais referentes, especificamente, à área de produção de jogos para celular: os testes de jogabilidade, usabilidade e compatibilidade. Qualidade e testes são tópicos de profundo relacionamento. Um jogo pode ser considerado a fusão de um software e um filme. Desta forma, para se conseguir um jogo de qualidade é preciso observar quais os quesitos de qualidade em um software e em um filme.

Do ponto de vista do jogo como um software, ele precisa conter as implementações que contemplam os requisitos funcionais e não-funcionais (definidos indiretamente pelos conceitos do jogo); precisa ser robusto, livre de falhas, ter disponibilidade e boa performance, dentre outras exigências. Neste sentido, um jogo deve passar por testes unitários, testes de integração, testes funcionais e não-funcionais (artefatos-padrão do RUP).

Observando um jogo pela perspectiva de um filme, ele precisa ser atrativo, divertido e ser um bom software de entretenimento. Desta forma, pode-se dizer que a criação de jogos é

definitivo para definição das exigências de um jogo, o que ajuda a sustentar o princípio de Processo Dirigido a Criação de jogos (seção 5.3.1). Neste sentido, um jogo deve passar por testes de jogabilidade, usabilidade e compatibilidade (artefatos incluídos no MGUP).

De acordo com o discutido na seção 3.3, o Criador de Jogos possui atuação fundamental no processo de evolução do *gameplay*. Isso é alcançado por meio de testes de jogabilidade realizados sobre protótipos físicos e em software. Estes testes fazem parte do *workflow* e processo de criação de jogos.

Os artefatos, papéis e atividades da disciplina de Testes que foram incluídos no MGUP foram apresentados nas seções 5.4.5 e 5.4.6.

5.5 Comparativo: MGUP e RUP

O MGUP é o resultado da investigação de adequação e aplicação do RUP ao desenvolvimento de jogos móveis. Vários elementos do processo foram discutidos ao longo deste capítulo. Esta seção apresenta um sumário comparativo do MGUP em relação ao seu processo base, o RUP.

A seção de artefatos e papéis lista os artefatos explicitamente citados na apresentação do MGUP, descartando-se aqueles que já constam na descrição do RUP e não são específicos para o desenvolvimento de jogos.

Elemento	Processo Unificado para Jogos Móveis	Processo Unificado da Rational
Princípios	• Orientado a *Criação de jogos* • Centrado na Arquitetura e no *core gameplay* • Iterativo e Incremental	• Orientado a Casos de Uso • Centrado em Arquitetura • Iterativo e Incremental
Fases	• Concepção • Pré-Produção • Produção • Pós-Produção	• Concepção • Elaboração • Construção • Transição

Disciplinas	• Criação de jogos
	• Requisitos
	• Análise & Design
	• Implementação
	  o Implementação de Software
	  o Implementação de Arte
	  o Implementação de Som
	• Teste (ênfase em jogabilidade)
	• Implantação
	• Gerenciamento de Configurações e Mudanças
	• Gerenciamento de Projetos
	• Ambiente
		• Modelagem de Negócio
		• Requisitos
		• Análise & Design
		• Implementação
		• Teste
		• Implantação
		• Gerenciamento de Configurações e Mudanças
		• Gerenciamento de Projetos
		• Ambiente

Elemento	Sumário de Artefatos do MGUP

Criação de jogos	• Especificação Conceitual do Jogo
	• Especificação do jogo
	  o Especificação de Níveis
	  o Especificação de Mecânicas do Jogo
	  o Especificação de Personagens
	  o Especificação da Física e I.A. do Jogo
	  o Especificação do Roteiro do Jogo
	  o Especificação de Padrões de Navegação, Telas e Fluxo

Software	• Especificação Técnica
	  o Arquitetura
	  o Soluções
	  o Problemas e Riscos
	• Especificação de Requisitos
	• Código

Arte	• Especificação de Arte • Arte Conceitual • Arte 2D • Modelos 3D • Personagens 3D • Animações 3D • Texturas • Story Board

Som	• Especificação de Mídia Sonora • Música • Efeitos Sonoros
Qualidade	• Plano de Testes de Jogabilidade • Plano de Testes de Usabilidade • Plano de Testes de Compatibilidade

Elemento	Papéis do MGUP (acrescentados ao RUP)
Planejamento	• Produtor (Gerente de Projetos) • Líder de Criação de Jogos • Líder de Música • Líder de Desenvolvimento • Líder de Arte • Líder de Qualidade • Planejador de Software • Arquiteto de Software
Criação de jogos	• Criador de Jogos • Criador de Níveis (*Level/Mission Designer*) • Criador de Personagens (*Character Designer*) • Criador de Mecânicas • Criador de Estória e Roteiro
Software	• Desenvolvedor o Desenvolvedor de Ferramentas o Desenvolvedor do Moto do Jogo o Desenvolvedor da Lógica do Jogo
Arte	• Artista Conceitual • Criador de Story Board • Artista 2D • Criador de Texturas • Modelador 3D • Modelador de Personagens • Animador 3D • Iluminador 3D
Som	• Músico • Criador de Efeitos Sonoros
Qualidade	• Técnico de Qualidade • Testador de Jogabilidade

5.6 Sumário

O MGUP foi o resultado de uma investigação da aplicação do Processo Unificado da Rational ao desenvolvimento de jogos móveis.

Como visto no capítulo 2, o RUP por si só é um *framework* de processo de software que precisa ser configurado às necessidades da organização ou do projeto. O desenvolvimento de jogos (capítulo 3) é repleto de peculiaridades por sua natureza multidisciplinar e também por suas semelhanças com o processo de criação de filmes. A realidade dos projetos de jogos móveis também apresenta diferenciações em relação às demais plataformas de entretenimento (capítulo 4).

O MGUP partiu de uma definição flexível, dinâmica e fortemente iterativa do Processo Unificado da Rational – RUP Lite –, indo ao encontro da área de desenvolvimento de jogos (e *criação de jogos*) pelos trilhos do RUP. O resultado dessa investigação é uma proposta de processo baseada no RUP (MGUP) mostrando um processo que contempla as etapas, os papéis, as atividades, os artefatos, as disciplinas e as práticas necessárias ao desenvolvimento de jogos móveis e, acima disso, à busca da produção de software de qualidade.

6 Experimentação e Resultados

6.1 Introdução

Este capítulo apresenta os estudos de casos realizados sobre projetos de desenvolvimento de jogos móveis em uma empresa de Pesquisa & Desenvolvimento: o Centro de Estudos e Sistemas Avançados do Recife – C.E.S.A.R. O C.E.S.A.R. é hoje a maior e mais importante empresa de inovação em tecnologia e computação nacional. Com um quadro de mais de 1.000 colaboradores, desenvolve projetos nos mais diversos domínios e plataformas.

Os estudos de casos foram realizados sobre projetos que apresentaram boa variação de características: o número de pessoas, os prazos, as tecnologias e plataformas utilizadas, os paradigmas visuais e, o gênero e a complexidade dos jogos. Essa variabilidade ajudou a demonstrar que o MGUP está de acordo com processos utilizados em projetos de desenvolvimento de diferentes jogos móveis.

Os estudos de caso realizados serviram como meio para validar, experimentalmente, a proposta apresentada.

6.2 Estudo de Caso

O C.E.S.A.R. é uma instituição de Inovação, Pesquisa e Desenvolvimento que possui dezenas de projetos nas mais variadas áreas. Em parceria com outros centros de pesquisa, universidades e empresas, desenvolve projetos de caráter inovador e comercial.

Durante o desenvolvimento deste trabalho, foi possível participar de projetos reais de jogos móveis. No total, foram desenvolvidos dez jogos móveis em quatro projetos (M1, M2, M3 e M4), num período de nove meses. As características destes projetos usados estão agrupados na Tabela 4.

No total, ocorreram três variações do processo de jogos usado. Eles são identificados por: Processo de Jogos Móveis I (PM1), Processo de Jogos Móveis II 2D (PM2 2D) e Processo de Jogos Móveis II 3D (PM2 3D). O PM 1 foi utilizado somente no primeiro projeto de jogos móveis, resultando em uma série de problemas. O PM2 2D é uma evolução do PM1, incorporando vários elementos do MGUP. Ele foi utilizado nos projetos M2 e M3. O PM2 3D é uma variação que acomoda alguns elementos de produção de jogos 3D e foi utilizado no projeto M4. Uma descrição detalhada de cada perfil de processo é feita nas sessões 6.3.1 e 6.3.2.

Os perfis de processo serão apresentados e discutidos em comparação à proposta do MGUP. Não foi possível obter uma documentação formal completa destes processos, assim como não foi possível divulgar detalhes dos projetos. Isso se deve ao sigilo exigido pela organização. A única documentação do processo que foi disponibilizada é um resumo do processo de jogos "genérico" do C.E.S.A.R.: o *Game Process*.

Os perfis de processo de jogos móveis apresentados são baseados no processo de jogos do C.E.S.A.R.. No Anexo III está um resumo deste processo, disponibilizado pela equipe de qualidade da organização. Por razões confidenciais, este foi o único documento oficial liberado.

Embora teoricamente os perfis de processo sejam baseados no *Game Process* do C.E.S.A.R., na prática eles apresentaram um pouco distante deste. Por esta razão, optou-se por uma descrição e análise sobre estes processos utilizados e ao invés de usar o *Game Process*. O documento foi anexado apenas para ilustrar o originário dos processos analisados.

Os perfis de processo foram selecionados por apresentarem diferenças significativas nas suas configurações, em especial com respeito à:

- Papéis envolvidos na equipe;
- Fases do processo;
- Artefatos Produzidos;

- Plataforma de desenvolvimento móvel (J2ME e Brew);
- Linguagem de programação (Java e C++);
- Ambiente de desenvolvimento (Eclipse e Visual C++);
- Paradigma visual (2D e 3D).

Tabela 4: Resumo dos projetos executados com os Perfis de Processo apresentados.

Dados dos Projetos do Estudo de Caso								
Projeto	**Gênero do Jogo**	**Paradigma de Jogo**	**Plataforma utilizada**	**Linguagem/ Ambiente (IDE)**	**Papéis**	**Pessoas**	**Tempo de Projeto**	**Perfil de Processo (seções 6.3.1 e 6.3.2)**
M1	Simulador de Vida	2D/SinglePlayer	J2ME	Java/Eclipse	**Compartilhado** 1x Game Designer 1x Engenheiro de Som **Dedicado a cada Jogo** 2x Desenvolvedor 1x Artista 2D	5	3 meses	J2ME 2D
	Corrida	2D/SinglePlayer	J2ME	Java/Eclipse				J2ME 2D
	Shooter Espacial	2D/SinglePlayer	J2ME	Java/Eclipse				J2ME 2D
M2	Esporte	2D/SinglePlayer	J2ME	Java/Eclipse	**Compartilhado** 1x Game Designer 1x Engenheiro de Som **Dedicado a cada Jogo** 2x Desenvolvedor 1x Artista 2D	5	3 meses	J2ME 2D
	Simulador de Negócios	2D/SinglePlayer	J2ME	Java/Eclipse				J2ME 2D
	Shooter Exploratório	2D/SinglePlayer	J2ME	Java/Eclipse				J2ME 2D
M3	Tabuleiro	2D/MultiPlayer	J2ME	Java/Eclipse	**Compartilhado** 1x Game Designer 1x Engenheiro de Som **Dedicado a cada Jogo** 3x Desenvolvedor 3x Artista 3D	5	3 meses	J2ME 2D
	Esporte/Tabuleiro	2D/MultiPlayer	J2ME	Java/Eclipse				J2ME 2D
M4	Arcade	3D/SinglePlayer	Brew	C++/Visual C++	**Compartilhado** 1x Game Designer 1x Engenheiro de Som **Dedicado a cada Jogo** 3x Desenvolvedor 3x Artista 3D	8	3 meses	Brew 3D
	Luta	3D/SinglePlayer	Brew	C++/Visual C++				Brew 3D

6.3 Processo de Experimentação

O processo de experimentação do MGUP foi definido com base em estudos de casos de projetos reais de jogos móveis. Algumas das dificuldades encontradas para a realização do estudo de caso foram:

- Somente dois locais do Brasil possuem empresas que desenvolvem jogos para celular: São Paulo e Recife. São Paulo possui baixa produção, com alguns fabricantes independentes. Recife é hoje o local considerado pólo de produção de jogos móveis.
- A melhor execução deste trabalho e a realização do estudo de caso só foram possíveis porque foi desenvolvido no Recife. O C.E.S.A.R., Centro de Estudos e Sistemas Avançados do Recife foi o verdadeiro laboratório deste trabalho, pois as experiências de processo foram realizadas e observadas na prática diária da equipe de desenvolvimento de jogos móveis.

Diferentes abordagens poderiam ser usadas para experimentações do MGUP. Quatro possibilidades de cenários foram levantadas:

- Aplicação do MGUP e captura de métricas: O cenário considerado ideal seria a obtenção de métricas durante a aplicação do MGUP em vários projetos reais. Desconsiderando as dificuldades organizacionais, um dos fatores negativos para essa alternativa seriam as variáveis a serem analisadas. Tempo, custo e re-trabalho, são variáveis que podem variar independentemente do processo usado, pois outros fatores podem levar ao sucesso ou a falha de um projeto: o mal planejamento, um escopo insuficientemente definido ou desenvolvedores sem a experiência e o conhecimento necessário para construir o software. Assim, considerou-se que, além das barreiras organizacionais – não foi possível conseguir uma organização que adotasse o processo de forma experimental –, a

experimentação baseada somente na observação de variáveis (como as citadas acima) poderia tornar o resultado final impreciso.

- Estudos de casos levantados de revisões bibliográficas: Existe pouca literatura sobre processo de software e jogos. Ainda menor é a que trata do desenvolvimento de jogos móveis. Além disso, não haveria formas de se garantir que as informações contidas na literatura fossem reais.
- Análise de documentos de processo de uma desenvolvedora de jogos móveis: Esta alternativa foi considerada viável do ponto de vista da experimentação, mas não foi possível obter documentações confidenciais de empresas do ramo.
- Análise comparativa usando literatura e estudo de casos reais: Essa alternativa foi considerava válida e viável. No momento da realização deste trabalho, o C.E.S.A.R. já havia desenvolvendo alguns jogos móveis em parceria com outras empresas, mas os projetos de estudo de caso foram os primeiros a serem produzidos completamente pela organização. Durante o desenvolvimento deste trabalho, foi possível participar de projetos de jogos móveis, usando plataformas, tecnologias e paradigmas diferentes. Isto contribuiu para uma boa amostragem de projetos. O processo usado em cada projeto é descrito e usado na avaliação do MGUP por meio de análise comparativa.

Abaixo são apresentadas tabelas que resumem os perfis de processo usados nos quatro projetos (M1, M2, M3 e M4), sob a ótica dos seus elementos – fases, papéis (e atividades) e artefatos. Em cada um, é realizada uma analise comparativa com a proposta do MGUP.

6.3.1 Processo de Jogos Móveis I (PM1)

6.3.1.1 Descrição do Processo

Informações do Processo		MGUP	
Processo:	Processo iterativo incremental baseado no RUP.		
Fases:	Concepção	✓	
		×	Pré-Produção
	Construção (2 iterações)	✓	Produção
	- Lançamento Alpha	✓	
	- Lançamento Beta	✓	
	Transição	✓	Pós-Produção
	- Lançamento Gold	✓	
Papéis:	Engenheiro de Sistemas	✓	Desenvolvedor de *Engine*, Lógica e Ferramentas.
		✓	Planejador e Arquiteto de Software.
		✓	Planejador de Testes e executor de Testes Funcionais, de Compatibilidade e de Jogabilidade.
	Artista 2D/Artista Conceitual	✓	
	Engenheiro de Som	✓	
	Criador de Jogos	✓	
	Líder de Projeto	✓	
	Gerente	✓	
	Gerente de Configuração e Versões	✓	Provindo do RUP
	Analista de Qualidade	✓	Provindo do RUP
Artefatos:	Plano de Projeto	✓	
	Plano de Desenvolvimento	✓	
	Lista de Riscos	✓	
	Especificação Conceitual do Jogo	✓	
	Especificação de Requisitos	✓	
	Especificação do jogo	✓	
	Plano de Testes (Funcional e Não-Funcional)	✓	
	Plano de Testes de Compatibilidade	✓	
	Questionário de Testes de Jogabilidade	✓	
	Planilha de Inspeção	✓	Provindo do RUP
	Matriz de Rastreabilidade	✓	Provindo do RUP (sem valor)
	Matriz de Testes de Regressão	✓	Provindo do RUP (sem valor)
		×	Especificação Técnica

6.3.1.2 Análise

O processo apresentado foi utilizado no primeiro projeto (M1). Em comparação com o MGUP, o PM1 não tem etapa de Pré-Produção e nem o artefato de Especificação Técnica. O papel do Engenheiro de Sistemas agrupa vários papéis de desenvolvimento de software no MGUP. Dois papéis e três artefatos provindos do RUP foram utilizados e sendo assim,

mantêm compatibilidade com o MGUP. Abaixo é apresentada uma análise sobre cada elemento do processo.

Fases:

- Concepção: Nesta fase, o projeto foi concebido (Plano de Projeto e Plano de Desenvolvimento), a Lista de Riscos foi identificada, o contrato foi firmado, o conceito inicial do jogo estabelecido e documentado no Especificação Conceitual do Jogo. O projeto recebeu aprovação antes do término da fase. Estiveram envolvidos nesta fase: Gerente, Analista de Qualidade e Criador de Jogos.

- Construção: A fase de construção foi reservada para a implementação dos jogos. Houve uma divisão em duas iterações, cada uma com um lançamento ao final (Alpha e Beta). Não houve definição clara do foco de cada lançamento. Alguns recursos foram colocados no primeiro e outros no segundo, não utilizando critério específico.

- Transição: Nesta fase os objetivos foram aplicar testes, fazer correções de *bugs* e pequenas melhorias e produzir documentos pendentes (Matriz de Rastreabilidade e Matriz de Testes de Regressão). Tipos de testes realizados: funcionais, compatibilidade e jogabilidade.

Papéis e Atividades:

Os papéis envolvidos no projeto estão contidos no MGUP. Pela pequena quantidade de integrantes, algumas pessoas assumiram vários papéis.

O Engenheiro de Sistemas agregou vários papéis do MGUP:

- Arquiteto de Software: foi responsável por definir e evoluir a arquitetura.
- Planejador de Software: foi responsável por elucidar e documentar os requisitos do jogo e também por planejar as tarefas a serem implementadas em cada lançamento.
- Desenvolvedor: foi responsável por implementar todo o software.

Na fase de Transição, também assumiu os papéis de:

- Técnico de Qualidade: criou e executou os Planos de Teste Funcionais, Não-Funcionais e de Compatibilidade.
- Testador Jogador: executou testes de jogabilidade informais, onde pôde fornecer *feebacks* para o Criador de Jogos.

O Líder de projeto atuou como assistente de produção, auxiliando o gerente em suas atividades e realizando interface com a equipe de desenvolvimento.

O papel do Artista 2D incluiu o do Artista Conceitual, criando todas as imagens conceituais, os protótipos de interface e as telas finais do jogo.

O Engenheiro de Som executou os papéis de Músico e Criador de Efeitos Sonoros.

O Criador de Jogos produziu a Especificação Conceitual do Jogo e o a Especificação do Jogo, realizando o Processo de Criação de jogos, descrito na seção 3.3.2.

Os papéis de Gerente, Gerente de Configuração e Versões e Analista de Qualidade foram executados de acordo com as definições do RUP.

Artefatos:

Os artefatos de produção são desconsiderados nesta análise e por esta razão não estão listados na descrição do projeto.

O Glossário do projeto e o Protótipo físico não foram produzidos. A falta do Glossário gerou problemas de comunicação.

O questionário de testes de jogabilidade foi usado para realizar os testes com jogadores convidados.

Alguns artefatos foram produzidos para satisfazer os requisitos do contrato, e não do projeto. Foram eles: Matriz de Rastreabilidade e Matriz de Testes de Regressão.

Fator Tempo e Tamanho da Equipe:

A equipe reduzida (cinco pessoas para cada jogo) e o curto prazo (três meses) para execução do projeto não permitiram que artefatos como Protótipo Físico e Protótipo de Software fossem produzidos. O Criador de Jogos teve de realizar experimentos no próprio jogo em produção, o que gerou re-trabalhos, tais como: re-implementação de mecânicas do jogo, telas e modificações na *engine* e, imagens que tiveram de ser refeitas para atender a novas especificações do jogo.

Avaliação do Processo:

Apesar de o processo utilizado representar um subconjunto do MGUP, a não utilização de alguns elementos trouxe conseqüências problemáticas:

1. Desentendimentos sem o uso do Glossário. Este problema foi agravado pelo fato deste ter sido o primeiro projeto da equipe.
2. Problemas em definição de foco de iterações e lançamentos: não foi empregado o princípio da Centralização no *core gameplay* (sessão 5.3.3).
3. Sobrecarga dos engenheiros por má definição de responsabilidades de papéis. Tarefas de criação de níveis, definição de comportamentos de NPC's, definição de roteiro do jogo e diagramação de telas foram realizadas pelos Engenheiros de Sistemas. Essa sobrecarga fez com que suas atividades e, consequentemente, o projeto, sofressem atrasos.
4. A falta de uma etapa de planejamento do software trouxe como conseqüência pouca reutilização: houveram muitos trabalhos replicados nos três jogos.
5. Também houve uma falta de padronização na arquitetura dos jogos por não haver um *framework* estável no começo do desenvolvimento, o que teria sido alvo de uma fase de Pré-Produção.
6. Ocorreram vários pontos não definidos no conceito do jogo, fatos que também são conseqüência da não realização da fase de Pré-Produção.
7. Houve problemas para medir tempo e esforço: falta de Especificação Técnica.
8. A dimensão das mudanças discutida no MGUP (sessão 5.4.2.1) não foi considerada, e mudanças cruciais foram executadas de forma tardia no projeto, acarretando em re-trabalhos e atrasos, além de constituírem fator de alto risco.

Como resultado final, pode-se afirmar que o esforço e a dedicação da equipe foram os grandes responsáveis pelos resultados do projeto. O processo foi falho em vários pontos, gerando atrasos, falhas de comunicação, re-trabalhos e alterações tardias (e bastante arriscadas) nos jogos.

6.3.2 Processo de Jogos Móveis II (PM2) 2D/3D

6.3.2.1 Descrição do Processo

Informações do Processo		MGUP	
Processo:	Processo iterativo incremental baseado no RUP.		
Fases:	Concepção	✓	
	Elaboração	✓	Pré-Produção
	Construção (2 iterações)	✓	Produção
	- Lançamento Alpha	✓	
	- Lançamento Beta	✓	
	Transição	✓	Pós-Produção
	- Lançamento Gold	✓	
Papéis:	Engenheiro de Sistemas	✓	Desenvolvedor de *Engine*, Lógica e Ferramentas.
		✓	Planejador e Arquiteto de Software.
		✓	Planejador de Testes e executor de Testes Funcionais, de Compatibilidade e de Jogabilidade.
	Artista 2D/Artista Conceitual	✓	
	Modelador 3D*	✓	
	Animador 3D*	✓	
	Engenheiro de Som	✓	
	Criador de Jogos	✓	
	Líder de Projeto	✓	
	Gerente	✓	
	Gerente de Configuração e Versões	✓	Provindo do RUP
	Analista de Qualidade	✓	Provindo do RUP
Artefatos:	Plano de Projeto	✓	
	Plano de Desenvolvimento	✓	
	Lista de Riscos	✓	
	Especificação Conceitual do Jogo	✓	
	Especificação de Requisitos	✓	
	Especificação de Arte*	✓	
	Lista de Tarefas	✓	Especificação Técnica
	Especificação do jogo	✓	
	Plano de Testes (Funcional e Não-Funcional)	✓	
	Plano de Testes de Compatibilidade	✓	
	Planilha de Inspeção	✓	Provindo do RUP
	Matriz de Rastreabilidade	✓	Provindo do RUP (sem valor)
	Matriz de Testes de Regressão	✓	Provindo do RUP (sem valor)

*** Elementos adicionados para os projetos 3D.**

6.3.2.2 Análise

Estes dois perfis de processo, PM2 2D e PM2 3D foram uma evolução do processo anterior, o PM1. A versão 3D do processo apenas adiciona dois papéis – Modelador e

Animador 3D – e um artefato – Especificação de Arte. O PM2 2D foi usado nos projetos M2 e M3. O projeto M4 usou a variação para jogos 3D (PM2 3D).

Vários elementos presentes no MGUP foram adicionados em relação em PM1 e trouxeram resultados bastante significativos. Também, fundamentos do MGUP passaram a ser usados, definindo melhores formas de execução dos projetos.

Sendo uma evolução de seu antecessor, não se faz necessário reapresentar a discussão sobre os elementos de processo comuns ao PM1 e PM2. Desta forma, somente os elementos adicionais são analisadas, com exceção daqueles que tenham sido modificados.

Fases:

- Concepção: A fase de concepção foi realizada da mesma forma que no projeto anterior: fechamento de contrato, plano de projeto, Plano de Desenvolvimento e conceito inicial do jogo.

- Elaboração: Neste projeto o processo sofreu o acréscimo da fase de planejamentos: a Pré-Produção do MGUP. Isto foi o resultado de sugestões de melhorias dadas no *Post-Morten* (lições aprendidas) do projeto anterior. Desta vez, houve um momento bem definido para discutir a arquitetura e as soluções de cada jogo, evoluir o *framework* existente, realizar reuniões de verificação de viabilidades junto com o Criador de Jogos, planejar de forma mais completa o jogo e seus elementos de arte e som.

- Construção: Novamente houve uma divisão em duas iterações, cada uma com um lançamento ao final (*Alpha* e *Beta*). O objetivo de cada lançamento desta fase foi o mesmo apresentado no MGUP para as macro-iterações da Produção: um primeiro lançamento contemplando o *core gameplay* (*Alpha*) e um segundo contendo o jogo completo (*Beta*).

- Transição: Na fase de transição deste projeto também foram aplicados testes, realizadas correções de defeitos e pequenas melhorias, e produção de documentos pendentes (também para cumprimento do processo).

Papéis, atividades e Artefatos:

Os papeis e artefatos adicionais em relação ao projeto anterior foram o de Modelador 3D e Animador 3D, devido ao fato de que os jogos desenvolvidos foram 3D. Respectivamente, eles produziram os artefatos Modelos 3D e Animações 3D. Novamente o Glossário não foi produzido e o mesmo problema de comunicação se repetiu embora em

menor intensidade, pois a cada projeto a equipe foi adquirindo mais experiência.

Durante a etapa de Elaboração foram produzidos a Especificação Técnica – uma Lista de Tarefas bastante detalhada – contendo o planejamento do software e as atividades a serem desenvolvidas em cada lançamento. Nos jogos 3D foi produzida uma Especificação de Arte, contendo imagens conceituais dos principais elementos dos jogos.

Avaliação do Processo:

Com a inclusão de novos elementos (provindos do MGUP) e conseqüente evolução do processo, o PM2 obteve resultados bastante satisfatórios, eliminando quase totalmente os problemas observados no projeto (e processo) anterior. Dentre as melhoras atingidas, pode-se citar:

1. Melhor planejamento do jogo, ocorrido durante a fase de Elaboração.
2. Houve um aceleramento da produção do jogo pelo uso de um *framework* bem definido (atividade realizada durante a Elaboração).
3. Com um jogo mais planejado, houve grande redução de atrasos gerados por indefinições de conceitos dos jogos.
4. Utilizou-se o modelo do MGUP de escopo para os dois lançamentos (*Alpha* e *Beta*). Isso ajudou o Criador de Jogos a delimitar melhor o escopo do jogo todo.
5. A diminuição de atrasos gerou um ganho de tempo que foi usado para aprimorar o *gameplay* dos jogos.
6. Com maior reutilização, menos atrasos, melhor delineação de escopo dos jogos e melhor planejamento, a equipes trabalharam de forma mais efetiva.

Os jogos superaram as expectativas internas e externas. Isso foi conseguido graças ao esforço e dedicação da equipe somada a um processo adequadamente definido e executado. O PM2 apresentou uma série de melhorias apresentadas no MGUP que trouxeram resultados bastante satisfatórios.

6.4 Análise final

Os estudos de caso realizados tiveram o propósito de contribuir para a experimentação da proposta do MGUP. Sendo uma proposta que usa o RUP como processo-base, o MGUP tem sua validação na literatura. No entanto, uma análise frente a projetos reais é necessária. Isso pode ser feito com os perfis de processo apresentados no estudo de caso.

O estudo mostrou que o primeiro processo usado (PM1) apresentou vários problemas. Com a inclusão e modificação de elementos do processo baseando-se na proposta do MGUP, o PM2 apresentou resultados bastante satisfatórios nos três projetos seguintes (M2, M3 e M4). Desta forma, o estudo de caso apresentou um processo fortemente baseado no MGUP que funcionou bem na execução de projetos com diferentes tamanhos, complexidades, tecnologias e paradigmas visuais.

Nenhum dos processos especificou a produção de um Glossário. Este se mostrou importante para evitar eventuais problemas de comunicação que ocorreram na equipe. Esses problemas eram mais freqüentes quando a equipe recebia novos integrantes.

Nem todos os artefatos, papéis e atividades do MGUP foram usados nos projetos. Provavelmente as principais razões para isto foram o prazo e o pessoal alocado. Estas características estão diretamente relacionadas ao escopo reduzido dos jogos desenvolvidos.

As diferenças dos quatro projetos enfatizam a independência do MGUP em relação à tecnologia, tamanho e conceito de jogo, e desta forma, seu uso como um processo de jogos móveis genérico.

Os processos usados nos projetos apresentaram total concordância com a proposta do MGUP. Eles podem ser considerados adaptações (especializações) do MGUP para um cenário específico de jogos móveis: projeto de curto prazo, com alocação de poucos recursos humanos e escopo de jogo bastante reduzido. Assim, o MGUP pode ser visto como um macro processo, a partir do qual derivam-se os processos usados nos projetos do estudo de caso.

Considerando que os projetos M2, M3 e M4 foram finalizados com sucesso (em boa parte graças ao processo usado), e também que seus processos são, conceitualmente, derivações do MGUP, pode-se afirmar que este é válido. Assim, considera-se que seus princípios podem ser aplicados a diferentes cenários de projetos de jogos móveis.

6.5 Resultados e Benefícios

Esta seção apresenta um quadro geral dos resultados obtidos perante o perfil de processo usado em cada um dos projetos realizados. As avaliações são feitas baseadas em questões-chave, indicativas de melhorias no processo, no projeto e no resultado final. Referências serão feitas para a Tabela 4, que apresenta um resumo dos projetos executados.

Nas avaliações seguintes, assume-se que os processos usados são instâncias do MGUP, uma vez que as primeiras versões dos dois perfis de processo apresentados (J2ME 2D e Brew 3D) , assim como todas as melhorias adicionadas no decorrer dos projetos estão previstas no MGUP.

O processo ajudou no desenvolvimento?

O processo serviu de linha guia para todo o ciclo de desenvolvimento. Nos três primeiros jogos (Projeto M1), não houve uma definição clara do objetivo de cada fase e lançamento dentro do processo. Como resultado, houve um deslocamento de tempo e o resultado final foi pior do que o esperado. Também, como não foi produzido um glossário, houveram alguns problemas de troca de informações dentro das equipes dos jogos.

Nos projetos M2, M3 e M4 foi realizada uma etapa de Pré-Produção. Por causa dela, o desenvolvimento foi acelerado e os jogos foram entregues no prazo esperado. A razão disto é que todo planejamento realizado na Pré-Produção e a construção/evolução da arquitetura executável utilizada em todos os jogos foi evoluída nesta etapa. Outro ponto melhorado no processo foi a definição clara dos objetos de cada fase e lançamento, o que permitiu um

planejamento preciso de quais requisitos deveriam ser implementados em cada lançamento.

O tempo de desenvolvimento foi reduzido?

Por questões contratuais, o objetivo dos projetos executados era desenvolver um determinado número de jogos em um tempo fixo. Assim, os conceitos dos jogos foram adaptados ao tempo existente. No entanto, houve ganho de tempo.

No primeiro projeto (M1), oito falhas (sessão 6.3.1.2) no processo acarretaram em atrasos (duas semanas). Além disso, os resultados em qualidade dos jogos não satisfizeram as expectativas da equipe. Nos projetos seguintes (M2, M3 e M4), a inclusão da fase informal de Pré-Produção, a produção/evolução de uma arquitetura executável reutilizável e a definição clara dos objetivos de cada lançamento permitiram às equipes entregar jogos que superaram as expectativas internas e externas.

Desta forma, pode-se afirmar que o tempo de desenvolvimento foi reduzido visto que as equipes puderam fazer ajustes e melhorias no *gameplay* dos jogos, sem precisar exceder o tempo planejado de cada projeto.

Os membros da equipe trabalharam de maneira mais efetiva?

A partir das melhorias empregadas no segundo projeto a equipe trabalhou de forma consideravelmente mais efetiva. Os resultados dos projetos M2, M3 e M4 comprovam isso: jogos melhores - jogabilidade mais refinada por meio de testes e melhoramentos, mais elementos implementados e melhor planejamento e resultados de arte – no mesmo tempo planejado para o primeiro projeto (3 meses).

O resultado foi satisfatório?

Jogos melhores no mesmo espaço de tempo, maior reutilização, melhor planejamento,

menor re-trabalho, equipes trabalhando de forma mais efetiva e superação de expectativas internas e externas: todas estas características confirmadas nas experimentações indicam que certamente o resultado foi satisfatório.

O cliente gostou dos resultados?

Embora os três primeiros jogos, resultantes do projeto M1, não satisfizeram as expectativas da equipe de desenvolvimento, o cliente ficou bastante satisfeito com todos os jogos resultantes de todos os quatro projetos. A prova disto é que cada novo projeto foi aberto com base na satisfação sobre os resultados do anterior. O cliente do projeto havia solicitado a realização dos projetos dos jogos por demanda. Este foi um mecanismo de segurança utilizado por ele para que, caso o resultados dos jogos não fossem satisfatórios, novos projetos não seriam confirmados. Os resultados dos quatro projetos foram tão satisfatórios que mais treze jogos foram solicitados.

Quais elementos do MGUP foram acrescentados e melhoraram o processo?

Os processos usados nos quatro projetos de jogos móveis podem ser considerados instâncias do MGUP, uma vez que conceitualmente estão contidos dentro de sua definição.

Os principais elementos do MGUP que representaram melhorias para o processo, e conseqüentemente, para o resultado final, foram: a inclusão de uma etapa de Pré-Produção, a definição clara dos objetivos de cada iteração, o planejamento de cada iteração, a produção de artefatos de controle que guiaram o desenvolvimento (especificações de software, arquitetura e listas de tarefas) e, por fim, o planejamento e execução dos testes de jogabilidade (não incluídos formalmente nos processos usados, mas realizados informalmente durante a execução dos projetos).

6.6 Conclusões do Trabalho

Os jogos móveis representam hoje o maior percentual de investimentos nacionais na área de jogos. O leque de possibilidades ainda é praticamente infinito, pois a cada dia, novas e mais sofisticadas tecnologias são incorporadas nos aparelhos de telefone celulares. A computação móvel definitivamente está em franco processo de crescimento e o que se têm hoje é apenas o início de algo grandioso.

O Brasil não despertou para o desenvolvimento de jogos em plataformas como Computadores Pessoais e Consoles de Videogames no momento certo, e isto custou anos de atraso em experiência e conhecimentos. Os jogos móveis representam a possibilidade do país entrar definitivamente na era do entretenimento digital como fornecedor de novas idéias, provedor de serviços e fábrica de jogos de qualidade. Tudo indica que este caminho é possível, e empresas como o C.E.S.A.R. e a Meantime Mobile Creations já deram o pontapé inicial que toda comunidade acadêmica e a indústria nacional precisavam.

O mercado internacional de jogos móveis está crescendo e logo, todas as gigantes fabricantes estrangeiras de jogos para outras plataformas demonstrarão interesse neste domínio. Electronic Arts, SEGA e Namco são três exemplos de multinacionais tradicionais do ramo de jogos que já começaram a desenvolver produtos neste setor. Não há momento mais certo do que este para se investir em tecnologia e ciência aplicada a jogos móveis.

Quando este trabalho foi idealizado, a intenção era verificar as possibilidades de uso do Processo Unificado da Rational para o desenvolvimento de jogos. O domínio de jogos móveis foi definido por representar uma nova perspectiva para o mercado, para a academia e para o Brasil. O resultado do trabalho apresenta muito mais do que uma simples resposta positiva ao uso de RUP em jogos: o MGUP – Processo Unificado para Jogos Móveis – representa uma proposta de solução de processo para o domínio de jogos móveis que pode ser especializado para atender as necessidades de projetos e organizações.

O MGUP não somente apresenta um ponto de vista do Processo Unificado da Rational voltado para um domínio específico. Mais que uma simples personalização, o MGUP é uma investigação que parte da essência e dos princípios do RUP, analisando quais são as melhores alternativas para remodelá-lo de forma a permitir as mesmas condições oferecidas por um processo de jogos genérico, acrescido das vantagens do modelo RUP. Como resultado, tem-se a documentação de um macro-processo de jogos móveis que pode ser aplicado a uma vasta gama de projetos. Ele foi instanciado conceitualmente na indústria e provou ter validade e eficácia.

No âmbito educacional, o MGUP cumpre com seu papel, servindo de instrumento incentivador a novas investidas no setor, uma vez que representa um estudo inovador no Brasil, abrindo um leque de novas possibilidades de pesquisas e produtos de software.

6.7 Trabalhos Futuros

6.7.1 Trabalhos Relacionados

As áreas envolvidas neste trabalho – Processo de Software, Desenvolvimento de Jogos e Aplicações Móveis – e os resultados obtidos constituem abertura para continuidade a novos trabalhos. Três trabalhos que poderiam refinar ou estender os resultados alcançados são: o estudo em torno da aplicação do MGUP a diversos cenários de jogos móveis; a definição de uma linguagem ou padrões para especificação da Concepção de um Jogo; e, a especificação de um padrão de Fábrica de Jogos de Celular.

A primeira sugestão de proposta é um estudo para aplicação e definição de configurações do MGUP para diferentes cenários de projeto de jogos de celular. Dentro do domínio de jogos móveis, variáveis de tempo de produção, número de integrantes da equipe, tamanho do jogo, paradigma visual (2D ou 3D) e número de jogadores, acabam definindo variações de perfis de projetos. Exemplos destes cenários são: jogos 2D simples, jogos 2D *multiplayer* simples, jogos 3D simples e jogos 2D completo, entre outros. A plataforma usada

no desenvolvimento do jogo também possui influência na forma como o processo precisa ser adaptado e aplicado ao projeto. São estas: J2ME, Brew, Flash Lite ou In-Fusio.

O aumento do interesse pelo desenvolvimento de jogos fez como que a área de c criação de jogos passasse a ser estudada sob várias óticas. Uma delas tem como objetivo encontrar uma forma mais padronizada de especificação do jogos. Este tipo de investigação pode fornecer recursos valiosos para aceleração e aumento da qualidade do processo de produção de jogos. Outra ótica de estudo são soluções reutilizáveis de conceitos de jogos – *Game Design Patterns* –, que também podem trazer excelentes resultados para a área.

Acima da definição de um processo de desenvolvimento, está a especificação de um modelo de Fábrica de Software. Acredita-se que com o resultado do presente trabalho e destas sugestões de extensões propostas seria possível formular um modelo de Ambiente de Produção de Jogos de Celular. Esta especificação poderia servir de "molde" para definição da forma de funcionamento de empresas produtoras de jogos móveis. Este modelo de Fábrica de Software proposto pode ser constituído, dentre outros elementos, de:

- Uma especificação de processo específico para jogos móveis: o MGUP.
- Um guia de aplicação deste processo para diferentes cenários de projetos de jogos móveis. Estes cenários podem ser definidos por variáveis de: nível de formalidade do processo, complexidade do jogo, tamanho da equipe e nível de inovação do conceito do jogo, dentre outras.
- Um gerador de modelos de artefatos para cada cenário de projeto.
- Um conjunto de soluções conhecidas para a criação do conceito de jogos móveis (*Game Design Patterns*).
- Uma *engine* ou *framework* de componentes reutilizáveis que disponibilize os serviços básicos de um jogo e provenha aceleração da produção, possibilitando aumento da qualidade do software.

- Um conjunto de soluções de projeto de software – Padrões de Projeto (*Degin Patterns*) – especializadas a partir das já conhecidas ou definidas especificamente para jogos móveis, visando também uma melhoria na qualidade e aceleração da produção.

6.7.2 Outros Trabalhos

Embora não sejam extensões ou especializações relacionadas ao processo de produção de jogos, outras sugestões de trabalhos futuros são mencionadas por estarem relacionadas às atividades executadas durante o desenvolvimento de jogos móveis. Também, essas sugestões têm relação direta com o tema de Fábrica de Jogos Móveis. São elas: arquitetura e *engine* de jogos e padrões de projeto de jogos.

A arquitetura e a *engine* do jogo abrem algumas possibilidades de estudo. A construção de uma *engine* baseada em componentes reutilizáveis é de grande valia para qualquer projeto de jogos móveis. A aceleração da produção e a melhoria na qualidade do software, e possivelmente do próprio jogo, são alguns dos resultados que se pode esperar de um trabalho neste sentido. Uma *engine* "multi-dispositivo" – que funciona em vários celulares – e "multi-plataforma" – que possa ser usado em J2ME e Brew, por exemplo – é um exemplo de ferramenta que pode contribui de forma bastante significativa com a pesquisa e o desenvolvimento de jogos móveis.

No âmbito da reutilização em arquitetura e soluções de softwares de jogos, uma investigação em torno da aplicação e especialização de padrões de projeto para o desenvolvimento de jogos móveis pode trazer bons resultados tanto para a indústria quanto para a academia. Além de uma melhoria significativa na qualidade e diminuição de esforço, este trabalho possibilitaria pesquisar novas soluções para problemas já conhecidos da produção de softwares de jogos.

As sugestões de trabalhos futuros apresentadas são apenas algumas das possibilidades mais significativas de pesquisa e desenvolvimento que podem trazer grandes resultados para uma evolução da forma pela qual os jogos são produzidos.

7 Referências Bibliográficas

7.1 Processo de Software

[BECK 2004]
BECK, Kent; ANDRES, Cynthia. **Extreme Programming Explained:** Embrace Change. 2. ed. Addison-Wesley, 2004.

[GIMENES 1994]
GIMENES, Itana M. de. S. **Uma Introdução ao Processo de Engenharia de Software.** Maringá, 1994.

[JACOBSON et. al. 1999]
JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, Jim, **Unified Software Development Process**. Addison-Wesley, 1999.

[KROLL & KRUCHTEN 2003]
KROLL, Per; KRUCHTEN, Philippe. **The Processo Unificado da Rational Made Easy**: A Practioner's Guide to the RUP. Addision-Wesley, 2003.

[KRUCHTEN 2000]
KRUCHTEN, Philippe. **The Processo Unificado da Rational:** An Introduction. 3.ed. Addison-Wesley, 2000.

[LARMAN 2003]
LARMAN, Craig. **Agile and Iterative Development:** A Manager's Guide. Addision Wesley, 2003.

[LARMAN 2001]
LARMAN, Craig. **Applying UML and Patterns:** An Introduction to Object-Oriented Analysis and Design and the Unified Process. 2.ed. Prentice Hall, 2001.

[POLLICE et. al. 2003]
POLLICE, Gary; LIZ, Augustine; LOWE, Chris; MADHUR, Jas. **Software Development for Small Teams:** a RUP-Centrix Aproach. Addison-Wesley, 2003.

[ROYCE 1998]
ROYCE, Walter, **Software Project Management:** A Unified Framework. Addison-Wesley, 1998.

7.2 Desenvolvimento de Jogos

[BARBAGALLO 2003]
BARBAGALLO, Ralph. **Wireless Game Development in C/C++ with BREW**. Wordware Publishing, 2003.

[BETHKE 2003]
BETHKE, Erik. **Game Development and Production**. Wordware Publishing, 2003.

[ESA 2005]
ESA (Entertainment Software Association). **Essential Facts about the Computer and Video Game Industry:** 2005 Sales, Demographics and Usage Data. 2005. Disponível em: <http://www.theesa.com/files/2005EssentialFacts.pdf>. Acesso em: fev. 2006.

[FLOOD 2003]
FLOOD, Kevin. **Game Unified Process (GUP)**. Disponível em: <http://www.gamedev.net/reference/articles/article1940.asp>. GameDev.net, 2003. Acesso em: fev. 2006.

[FLYNT 2004]
FLYNT, John P; SALEM, Omar. **Software Engineering for Game Developers**. Premier Press, 2004.

[FULLERTON 2004]
FULLERTON, Tracy; SWAIN, Christopher; HOFFMAN, STEVEN. **Game Design Workshop:** Designing, Prototyping and Playtesting games. Gama Network: CMP Books, 2004.

[GAMASUTRA 2005]
Gamasutra. **Game Development Articles**. CMP Game Group. Disponível em: <http://www.gamasutra.com/php-bin/article_display.php>. Acesso em: fev. 2005.

[GAMEDEV 2005]
GameDev.net. **Game Development Articles**. Disponível em: <http://www.gamedev.net/reference>. Acesso em: fev. 2005.

[LAM 2004]
LAM, Jason. **J2ME Game Development with MIDP 2.0**. Disponível em: <http://www.jasonlam604.com/books.php>, 2004. Acesso em: set. 2004.

[PEDERSEN 2003]
PEDERSEN, Roger E. **Game Design Foundations**. Wordware Publishing, 2003.

[ROLLINGS 2003]
ROLLINGS, Andrew; MORRIS, Dave. **Game Architecture and Design**. New Riders, 2003

[ROUSE 2000]
ROUSE, Richard; RYBCZYK, Mark Louis. **Game Design:** Theory & Practice. Wordware Publishing, 2000.

[RUCKER 2002]
RUCKER, Rudy. **Software Engineering and Computer Games**. Addison Wesley, 2002.

[WELLS 2004]
WELLS, Martin J. **J2ME Game Programming**. Premier Press, 2004.

7.3 Tecnologias e Ferramentas

[J2ME 2005]
J2ME. **J2ME:** Java 2 Micro Edition. Sun Microsystems, 1994-2006. Disponível em: <http://java.sun.com/j2me>. Acesso em: fev. 2006.

[BREW 2005]
BREW. **Brew:** Binary Runtime Environment for Wireless. Qualcomm Incorporated, 2002-2006. Disponível em: <http://brew.qualcomm.com/brew/en>. Acesso em: fev. 2006.

[FLASH 2005]
FLASH. **Macromedia Flash.** Adobe Systems Incorporated. Disponível em: < http://www.adobe.com/products/flash/flashpro/?promoid=BINT >. Acesso em: fev. 2006.

[FLASH LITE 2005]
FLASH LITE. **Macromedia Flash Lite.** Adobe Systems Incorporated. Disponível em: <http://www.adobe.com/products/flashlite>. Acesso em: fev. 2006.

[GAME MAKER 2005]
OVERMARS, Mark. **Game Maker.** Mark Overmars. Disponível em: <http://www.gamemaker.nl>. Acesso em: fev. 2006.

[IN-FUSIO 2005]
IN-FUSIO**. Ege System**. In-Fusio Mobile Games. Disponível em: <http://developer.in-fusio.com>. Acesso em: fev. 2006.

[MOPHUN 2005]
MOPHUN. **Mophun.** Synergenix Interactive, 2004. Disponível em: <http://www.mophun.com>. Acesso em: fev. 2006.

[VIRTOOLS 2005]
VIRTOOLS. **Virtools Developer Suite.** Virtools: The Behavior Company, 2004. Disponível em: <http://www.virtools.com>. Acesso em: fev. 2006.

7.4 Exemplos de Artefatos

[CROSSFIRE 2006]
CROSSFIRE. **TO CrossFire**. Disponível em: <http://www.to-crossfire.net>. Acesso em: mar. 2006.

[DEVILBEAR 2006]
DEVIL BEAR. **Devil Bear Entertainment**. Disponível em: <http://www.devilbear.net. Acesso em: mar. 2006.

[HALFLIFE2 2005]
HALFLIFE2. **Half Life 2**. Valve Corporation, 2005. Disponível em: <http://half-life2.com>. Acesso em: mar. 2005.

[MEGAMAN7 1995]
MEGAMAN7. **Megaman VII**. Capcom Co., 1995. Disponível em: <http://www.capcom.com/megaman>. Acesso em: mar. 2006.

[STORYBOARDARTIST 2006]
STORYBOARDARTIST. **The Story Board Artist**. Disponível em: <http://www.thestoryboardartist.com>. Acesso em: mar. 2006.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo