UNIVERSIDADE DE CAXIAS DO SUL
CENTRO DE COMPUTAÇÃO E TECNOLOGIA DA INFORMAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Projeto de Diplomação

Story Teller: Um jogo para criação de narrativas interativas

Bruno Lorandi Pagno
brunopagno@gmail.com

Elisa Boff
eboff@ucs.br

Tipo de Trabalho: pesquisa e implementação

Caxias do Sul, 6 de Dezembro de 2011.

UNIVERSIDADE DE CAXIAS DO SUL
CENTRO DE COMPUTAÇÃO E TECNOLOGIA DA INFORMAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Story Teller: Um jogo para criação de narrativas interativas

Trabalho de Conclusão de Curso para obtenção do Grau de Bacharel em Ciência da Computação da Universidade de Caxias do Sul.

Bruno Lorandi Pagno
brunopagno@gmail.com

Elisa Boff
eboff@ucs.br

Caxias do Sul, 6 de Dezembro de 2011.

Agradecimentos

Há quatro pessoas em especial que eu desejo agradecer. Meus pais, Sérgio e Nilva, por terem tornado possível que eu fizesse este curso e por terem me incentivado a estudar e aprender desde que eu era pequeno. Minha irmãzinha, Anne, que me ajudou muitas vezes com suas habilidades de ilustração e que acompanhou todo o desenvolvimento e todos os problemas junto comigo. E minha doce namorada, Débora, por ter estado comigo, por ter sorrido todas as vezes que eu precisei e por ter me feito ir muito além do que eu imaginava, sem me deixar desistir nunca, e sem me deixar me contentar com pouco. Além destes devo fazer um agradecimento extra à minha orientadora, Elisa, por ter se esforçado para tornar este trabalho melhor, e por ter me dado ideias e apoio o tempo todo. Por fim, um obrigado a todos os que me ajudaram a fazer os testes do projeto.

Resumo

Este trabalho mostra como jogos são uma forma interessante para se contar histórias. Narrativas e jogos andam juntos desde o surgimento dos videogames. Conceitos como narrativas em jogos eletrônicos, tecnologias de desenvolvimento de aplicações móveis e desenvolvimento de jogos são abordados neste projeto. O texto apresenta, também, a modelagem e proposta de implementação do jogo **Story Teller,** capaz de permitir aos jogadores a criação de uma pequena narrativa, através de um *smartphone*.

Palavras-chave: Narrativas interativas, desenvolvimento de jogos, aplicativos móveis.

Abstract

This work shows how games are an interesting way to tell stories. Narrations and games walk together since video games were created. Concepts as narratives in games, mobile applications development technology and game development are analyzed in this project. It's, also, proposed the creation of the game **Story Teller**, that will allow players to create a small narrative though a mobile phone.

Keywords: Interactive narratives, game development, mobile applications.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

Sigla	Significado em Português	Significado em Inglês
API	Interface de programação de aplicativos	Application programming interface
GPRS	Serviço de rádio de pacote geral	General packet radio service
NPC	Personagem não-jogador	Non-player character
OpenGL	Biblioteca gráfica livre	Open graphics library
OpenGL ES	Biblioteca gráfica livre para sistemas embarcados	Open graphics library for embedded systems
RPG	Jogo de interpretação de personagem	Role-playing game
SDK	Kit de desenvolvimento de software	Software development kit
SDL	Simples camada de mídia direta	Simple directmedia layer
UML	Linguagem unificada de modelagem	Unified modeling language
IDE	Ambiente Integrado de Desenvolvimento	Integrated Development Environment

Sumário

1. Introdução

Criatividade é a palavra chave quando se fala de atividades lúdicas. Brincadeiras e jogos infantis, por mais que sejam parecidos, sempre possuem novas possibilidades e caminhos a cada vez que se brinca ou joga. Um rabisco a mais num desenho, ou uma proteção a menos numa brincadeira de pega-pega fazem diferença a ponto de tornar a atividade divertida ou não.

Esta ideia é seguida pela maioria dos jogos eletrônicos. Desde os primeiros *video games* até os mais atuais, o espírito de desafio e a criatividade de seus criadores é o que os torna divertidos. Muitos tipos de jogos, estilos artísticos e modelos de narrativa foram desenvolvidos desde então, e da mesma forma foram criados inúmeros jogos, cada um com suas peculiaridades.

Mesmo no princípio as narrativas estavam presentes na grande maioria dos jogos digitais. Isso pode ser percebido verificando a história dos games. Segundo o *Guiness World Records* (2009), *Colossal Cave Adventure* (1976) foi criado antes dos clássicos *PacMan* (1980) e *Tetris* (1984), e já apresentava elementos de narrativa. Como era jogado em modo texto, seu único recurso era contar histórias. Cabia ao jogador tomar decisões do que fazer em cada situação apresentada. Ip (2010) também afirma que mesmo em jogos onde não havia ênfase na história existiam enredos simples, mas atraentes, com o objetivo de chamar a atenção do público. Os jogadores poderiam entender que a história não era um diferencial, mas a sua ausência seria percebida e ruim para o produto final.

Dentre os vários gêneros de jogos, os chamados RPGs (*Role-Playing Games*), bem como os *games* de aventura, são os que possuem o maior foco na narrativa (Novak 2010). A importância da narrativa, em alguns desses jogos, é tão grande que o propósito do jogo torna-se quase unicamente apresentar a história.

Parte da diversão está em viver as histórias propostas pelos criadores. O jogador faz parte da história que está jogando, e a capacidade de influenciá-la pode ser muito importante. Isso se torna mais evidente através do seguinte exemplo:

> "Sem dúvida, um dos fatores que atrai mais jogadores para as narrativas que os games oferecem é a possibilidade de escolha do percurso narrativo que extrapola, muitas vezes, a lógica linear comum dos formatos narrativos convencionais. Outro fator importante é que a narrativa nos jogos não são

> simplesmente compreendidas e interpretadas pelos jogadores, mas sim vivenciadas e significadas através da transformação de jogadores em personagens" (Beatriz et al., 2010).

Existem vários títulos que oferecem histórias interativas, onde o jogador participa diretamente na sequência dos acontecimentos da narrativa, como por exemplo *inFAMOUS* (2009), e *Chrono Cross* (2000). Em ambos os casos, o jogo sofre alterações de roteiro de acordo com a tomada de decisões do jogador. Porém, mesmo que estes jogos ofereçam caminhos alternativos, suas narrativas ainda são bastante rígidas. As opções do jogador não são o suficiente para apresentar histórias realmente alternativas.

Outro fator importante é a forma como se dá a interação com o jogo, tanto no aspecto físico (hardware), quanto na interface gráfica. "A Interatividade é um dos principais conceitos usados para caracterizar o processamento de informação através das novas mídias" (Koolstra e Bos, 2009, tradução do autor). Uma narrativa, por natureza, pode ser apresentada em forma escrita. Os jogos trouxeram a parte visual para primeiro plano, e, especialmente nos últimos anos, os consoles e as novas plataformas de jogos apresentaram novas maneiras de se interagir com uma história. Os RPGs de última geração como *Final Fantasy XIII* (2010) e *White Knight Chronicles* (2010), possuem uma grande quantidade de diálogos dublados. No jogo *Phoenix Wright* (2001), parte da interação depende do microfone e, consequentemente da fala do jogador.

Nos jogos, além da interação nas narrativas, existe a capacidade de criação de conteúdo, sejam mapas, cenários, personagens ou *quests*[1]. Alguns jogos se tornaram famosos justamente por esta qualidade, como é o caso de *Little Big Planet* (2008) e *ModNation Racers* (2010). A criação de conteúdo torna possível um nível maior de controle sobre os eventos do jogo por parte dos jogadores, e isso é um grande atrativo para muitos. O ponto negativo, porém, é a limitação da criação, que muitas vezes se resume a novos níveis, ou novos mapas para o jogo em questão. Este conteúdo, normalmente, não traz nenhuma adição para a história do jogo, e fica como um extra. Faz parte da experiência do jogo, mas não faz parte da narrativa.

---

[1] Palavra em inglês que significa "aventura". Comumente utilizada em jogos

Partindo desta situação, onde a maioria dos jogos da atualidade não tem essas características, este trabalho resultará em um jogo que dê ao jogador a possibilidade de construir uma narrativa com ambientação e conteúdo, e também que permita a ele visualizar, e, especialmente, interagir com a narrativa criada. O jogador, por sua vez, deverá produzir uma narrativa com as ferramentas dadas.

Este texto está organizado da seguinte forma:

Capítulo 2 - Narrativas em jogos eletrônicos:

Conceitos de narrativas e conceitos dos jogos atuais. Como narrativas e jogos andam juntos. Como são as narrativas de jogos, e como é dada a interação com elas. Breve análise de roteiros de jogos reais.

Capítulo 3 - Tecnologias para aplicações em dispositivos móveis:

Conceitos sobre a criação de aplicações para plataformas móveis e de jogos. Conceituação e análise de motores de jogo.

Capítulo 4 - Definição do jogo:

Explicação do que é jogo proposto neste trabalho. Conceitos, modelos e estruturação de como se dá o desenvolvimento.

Capítulo 5 - Processo de produção do jogo:

Apresentação do processo de produção do software.

Capítulo 6 - Resultados obtidos:

Apresentação dos resultados obtidos durante a produção do jogo.

2. Narrativas em jogos eletrônicos

Narrativas e jogos andam juntos desde o surgimento dos primeiros *video games*. Desde então, os jogos foram se tornando cada vez mais, uma forma de se contar histórias. A união desses conceitos segue o mesmo caminho do teatro e do cinema, porém os jogos ainda são um tipo de mídia relativamente nova e incipiente.

Para tornar mais claro o entendimento, primeiro será feito uma breve descrição do que é uma narrativa e seus principais conceitos para depois aplicarmos essas ideias ao mundo dos *video games*. Também será discutido de forma rápida um pouco sobre o cenário atual da indústria dos jogos eletrônicos.

2.1 Narrativas

"Um texto narrativo é um texto no qual um agente narrativo conta uma história" (BAL, 1999, tradução do autor). Apesar de parecer uma definição genérica, ela representa bem o que é, não só um texto narrativo, mas uma narrativa propriamente dita, pois este é um conceito antigo, presente na história humana desde sempre. Segundo Herman e Vervaeck (2005, tradução do autor) "Não houve nenhum período ou sociedade sem narrativas. Um bom número de pensadores contemporâneos ainda adicionam, qualquer coisa que se diga ou pense sobre um certo período ou lugar se torna uma narrativa." De forma similar, Chris Crawford (2005) indica que todas as culturas tem histórias, e que as histórias são o principal veículo de transmissão de informação cultural. A conceituação de uma narrativa parece ser da natureza do ser humano pois vivemos e experimentamos isso durante toda nossa existência, mas por outro lado, se analisarmos uma narrativa de forma mais didática encontramos algumas ideias mais minuciosas que podem dar uma clareza maior ao tipo de conhecimento que estamos tratando.

O primeiro conceito importante de uma narrativa é o ponto de vista. Uma narrativa pode ser contada em primeira ou terceira pessoa. Em primeira pessoa, o narrador é um personagem presente na história, enquanto em terceira pessoa ele é apenas um observador. Shokouhi, Daram e Sabah (2011) afirmam que o ponto de vista é constituído por muito mais do que uma posição física do narrador, mas também de suas opiniões políticas e posição social, assim como outros fatores

psicológicos. Por culpa desses fatores, a história pode ficar distorcida, especialmente no caso da narração ser feita em primeira pessoa. É claro que, mesmo em terceira pessoa, as opiniões do narrador influenciam a forma como a história é contada, mas em oposição ao primeiro caso, o texto apresenta os acontecimentos como fatos, e não apenas opiniões. Por exemplo, se a frase “o jogador venceu por sorte” for dita por um narrador em terceira pessoa, isso pode ser considerado fato, porém se o narrador estiver na primeira pessoa representa apenas opinião do personagem.

O segundo item importante de uma narrativa é a linearidade da história. Ela pode ser contada seguindo os acontecimentos cronologicamente, ou invertendo fatos. Muitas mídias famosas iniciam a narrativa num ponto, e em seguida mostram todos os acontecimentos que se passam antes desse momento específico, como, por exemplo, o jogo *Final Fantasy Tactics* (1999). Em outros casos a narrativa mostra uma história com uma linearidade totalmente alternativa, indo do final para o começo, ou, até mesmo com narrativas paralelas. Estes recursos não costumam ser muito utilizados porque adicionam um certo nível de complexidade nas histórias que podem acabar por confundir o espectador.

2.2 Jogos eletrônicos

Jogos possuem uma forma lúdica por natureza. Desde os primeiros *video games*, até os mais atuais, poucos são os exemplos de jogos que não são destinados ao lazer. Segundo Novak (2010) “Nas bases militares, *games* eletromecânicos eram fornecidos aos recrutas para distraí-los dos rigores do treinamento básico”. Esse objetivo tão simples se desenvolveu e passou a mover grandes quantidades de dinheiro, e, consequentemente passou a gerar jogos com altos níveis de complexidade.

A partir de então, surgiram múltiplas plataformas, dedicadas ou não a jogos. Inicialmente apareceram os fliperamas, máquinas de jogos simples como *Frogger* (1981). Então foram criados os consoles, e após, os consoles portáteis. Estes se tornaram uma plataforma de jogos muito popular. Consalvo (2006) indica que apesar da popularidade dos fliperamas, as grandes empresas da época acabaram dando preferência para os consoles. Provavelmente este fator influenciou no crescimento

dos consoles, que até hoje são bastante populares. Durante o percurso, também surgiram jogos para outras plataformas não específicas para *video games* como computadores, e até mesmo *DVD Players*. Atualmente os celulares tem recebido bastante destaque como plataforma de jogos.

O cenário atual da indústria propicia várias aplicações para jogos. O principal, como foi descrito, é o entretenimento, mas também fala-se muito de construção de comunidades, educação, treinamento e marketing, quando se trata deste tipo de mídia.

2.3 Narrativas e o mundo dos *video games*

Aconteceu de forma natural, a união das narrativas e dos video games. Primeiro surgiram os jogos de aventura em modo texto. O jogador lia uma descrição da cena em que se encontrava e inseria uma ação. Depois começaram a surgir jogos de aventura do estilo *point and click*, que trouxeram, além de uma arte visual mais bem elaborada, personagens para o jogador controlar. Eventualmente surgiram os RPGs, que são, atualmente, os jogos que possuem o maior foco em histórias.

Os jogos tem uma grande popularidade como contadores de histórias por culpa de sua natureza interativa. Segundo Lebowitz e Klug (2011, tradução do autor) "[...] a habilidade do jogador de interagir e afetar a história criou muitos novos tipos de histórias que são difíceis, senão impossíveis de ser representadas em outras mídias". É impossível outra mídia ser tão dinâmica quanto um jogo, pois, nele, o espectador interage diretamente com a história.

Existem vários gêneros e classificações para os jogos atuais, cada um com suas peculiaridades. Novak (2010) apresenta alguns tipos. Jogos de ação que possuem um ritmo acelerado, ou de aventura que oferecem ao jogador a possibilidade de explorar e descobrir um mundo novo. RPGs que contam uma história, e jogos de simulação e esportes que tentam nos dar o maior realismo dentro do virtual. Jogos de estratégia que dão controle de civilizações para o jogador, ou de cassino que permitem que o jogador aposte sem perder seu dinheiro. Todos esses gêneros podem ter um roteiro e uma história servido de base para o jogo.

Dependendo do tipo de jogo o contexto de história fica mais fortemente presente ou não, porém a presença de uma narrativa consistente o suficiente para ser emocionalmente envolvente, pode ser de vital importância para o sucesso do produto final. Lebowitz e Klug (2011, tradução do auto) afirmam “jogos não ‘necessitam’ de histórias. Se um jogo for divertido, pessoas jogarão. Apesar disso, praticamente qualquer jogo pode ser melhorado por uma boa história”. Se analisarmos, por exemplo, jogos de tiro em primeira pessoa, a princípio, uma história é pouco relevante, e não precisa ser muito elaborada. Porém, os jogos de maior sucesso do estilo, são os que representam alguma guerra que aconteceu no mundo real, e consequentemente se baseiam fortemente em uma história. O envolvimento do jogador é muito maior quando há essa motivação para o jogo.

Há outros tipos de jogos onde, a princípio, não faz sentido construir uma narrativa, como os jogos de cassino. Porém, se existe a possibilidade, os desenvolvedores aproveitam a oportunidade. Lesnovski (2011) apresenta uma forma de narrativa no jogo *The Sims 3* (2009), mostrando que até mesmo em jogos de simulação é possível criar histórias. O caso desse jogo é ainda mais interessante, pois as narrativas provindas dele são construídas pelo próprio jogador. O jogo simula a vida normal de uma pessoa, portanto toda a história do personagem é o jogador que constrói, como, por exemplo, o ato de conseguir um emprego, comprar uma TV muito grande, ou até mesmo arquitetar um novo andar para a própria casa.

Porém, como foi mencionado, de todos os tipos de jogos, o que mais possui foco em histórias são os RPGs. O próprio nome do gênero já indica que o jogo é feito para “interpretação de personagens”, portanto costuma ter um roteiro forte. O trecho a seguir indica o surgimento do gênero:

> “O RPG é uma categoria de jogos que surgiu na década de 70 e descende dos jogos de simulação de guerra. Distingue-se da maioria dos jogos por não haver condição de vitória, isto é, ao final de uma partida não existe jogador vencedor ou perdedor. A versão eletrônica do RPG surgiu quase na mesma época e possui mais visibilidade que sua versão original, chamada RPG “de mesa”. Destaca-se por ser uma das categorias de jogos digitais (junto com a categoria Aventura) que mais enfocam o lado narrativo do jogo, valorizando elementos dramáticos para causar deslumbramento” (Araújo e Ramalho, 2011).

2.4 As narrativas e a capacidade de interação

As histórias dos jogos costumam ser contados como uma narrativa tradicional. Mas, essas narrativas podem possuir algumas peculiaridades que são atribuídas a este tipo de mídia.

Segundo Lebowitz e Klug (2011), as formas mais comuns para histórias em jogos são:

- Múltiplos finais: dependendo das escolhas do jogador, a história apresenta dois ou mais tipos diferentes de desfechos.
- Caminhos ramificados: o jogador decide por seguir um de vários caminhos (a decisão pode não estar explícita).
- Final aberto: alguns jogos apresentam finais que deixam acontecimentos sem explicação ou sem final definido.

Embora estes três itens sejam explorados, as narrativas em sua forma natural, sem estes atributos especiais, continuam sendo as mais utilizadas. Principalmente pois há um esforço muito grande para se construir estes detalhes a mais, e muitas empresas não estão dispostas a investir em um recurso que pode não dar tanto retorno.

Estes tipos de histórias dão uma falsa sensação de poder ao jogador. Apesar de tudo o que um jogo pode oferecer, indo muito além de outras mídias “estáticas”, eles são limitados. Segundo Lebowitz e Klug (2011, tradução do autor) “a coisa mais importante a se perceber é que em qualquer jogo, um jogador pode fazer somente as coisas que os *designers*, programadores, escritores e outros criadores decidiram que poderiam quando estavam construindo o jogo”. Araújo e Ramalho (2011) também destacam esta limitação com algumas comparações do RPG de mesa e o RPG digital, mostrando como o digital fica limitado ao jogo, enquanto o de mesa possui infinitas possibilidades.

Jogadores interagem com histórias estáticas, ou pouco dinâmicas. A única possibilidade de interação sem, ou com menos limitações, seria ter uma equipe criando cenários e possibilidades em reação às ações do jogador. Infelizmente esta

é uma ideia impraticável visto que a produção de conteúdo para o jogo é muito mais demorada que a velocidade que o jogador o joga.

2.5 Roteiros reais

Desenvolvido em 1998 pela *Square*, *Final Fantasy Tactics* (1998) é considerado um clássico por vários jogadores. Um dos principais motivos é a sua história. Seu gênero é RPG tático.

A história do jogo é dividida em quatro capítulos, todos eles se passando na mesma localidade fictícia, chamada Ivalice. O personagem principal é Ramza Beuolve. Cada capítulo apresenta alguns personagens, algumas batalhas e diálogos sobre os acontecimentos. Ramza é um personagem que pouco entende tudo o que está acontecendo a sua volta, e o jogador vai aprendendo junto ao protagonista sobre a história do jogo. Segue um trecho da história do jogo:

> "Um ano se passou desde que Ivalice foi derrotada na "guerra dos 50 anos". O príncipe, de apenas 2 anos, era jovem demais para acender ao trono que seu pai o deixara. Dois tios do príncipe eram os possíveis candidatos a regentes. A disputa por poder de ambos acabou gerando muitos conflitos dentro do país, que já estava quebrado devido a guerra" (Square, 2011, tradução do autor).

O primeiro item percebido é o ponto de vista. Há uma diferença entre narração em primeira ou terceira pessoa, em textos e em jogos. No caso do *Final Fantasy Tactics* (1998), apesar de acompanharmos um personagem durante todo o jogo, pode-se considerar que a história é contada em terceira pessoa pois existe a figura de um narrador.

Quanto à linearidade, o jogo inicia em um ponto crítico, que seria equivalente ao início do segundo capítulo. Logo após uma pequena cena, a narrativa volta para um ponto anterior na linha do tempo e então se dá o primeiro capítulo.

Apesar de mostrar algumas cenas fora da ordem cronológica, *Final Fantasy Tactics* (1998) tem uma narrativa tradicional, sem múltiplos finais, e sem possibilidades de múltiplos caminhos. Mesmo assim, sua história é consistente e apreciada por muitos fãs.

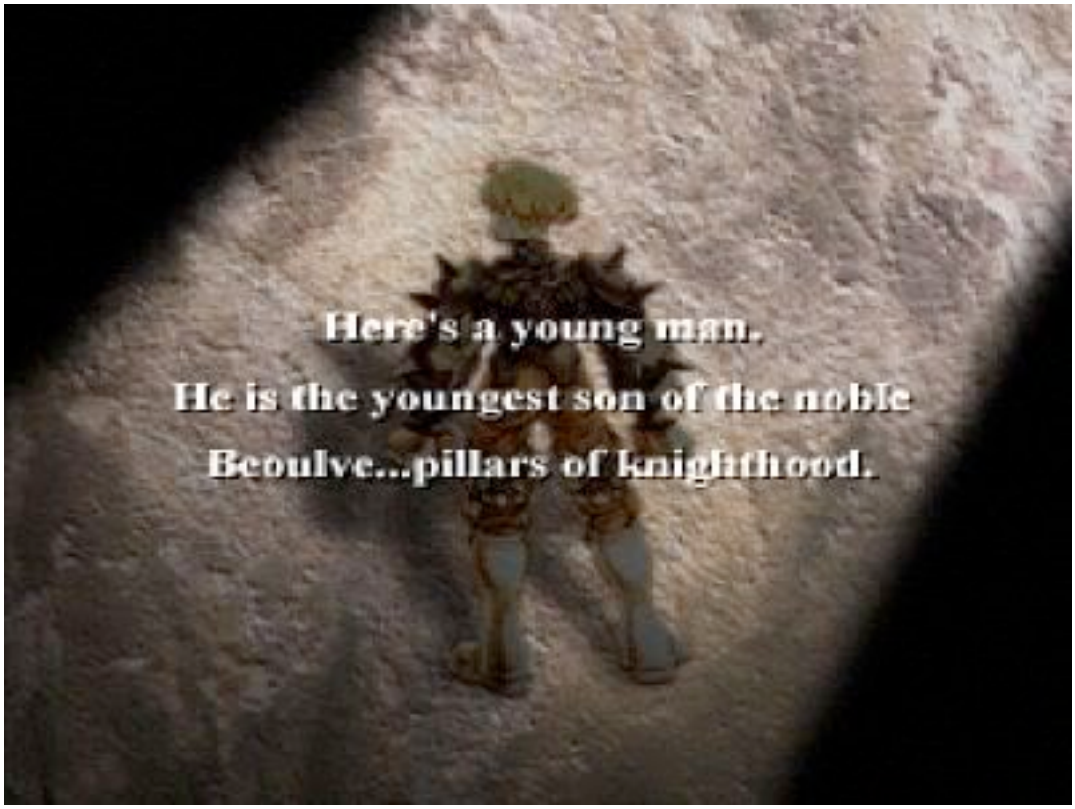


Figura 1 - Abertura de Final Fantasy Tactics.
Um narrador está contando a história ao jogador.

De outro ponto de vista, pode-se analisar *inFAMOUS* (2009). O jogo é relativamente novo, e portanto possui muito mais recursos do que o *Final Fantasy Tactics* (1998). Porém serão destacados apenas alguns aspectos da narrativa. Segue um trecho da história do jogo:

> "Cole era um cara normal. Entregando pacotes em sua bicicleta surrada, levando qualquer tipo de coisa para seu chefe... pois é, ele era, realmente, um ninguém. Mas então '*the blast*' rasgou Empire City ao meio e seu mundo se transformou num piscar de olhos. De repente Cole possuía poderes que ele jamais poderia ter imaginado. De repente ele era um fora da lei. De repente ele era um tipo de super herói" (InfamousTheGame, 2011, tradução do autor).

inFAMOUS é um jogo de mundo aberto e dá a possibilidade ao jogador de explorar uma cidade inteira. Cole é o personagem principal, e também é o responsável por contar a história. Ele mesmo narra seus passos como se estivesse contando fatos do passado para alguém, por isso é considerado uma narrativa em primeira pessoa.

O jogo segue uma história totalmente linear, desde o ponto inicial até o ponto final. Porém, o grande diferencial deste título é a possibilidade de escolha do caminho até o desfecho. *inFAMOUS* permite que o jogador decida se Cole será um herói ou um tipo de tirano. Em vários pontos do jogo é permitido que se decida fazer uma ação boa ou má, criando uma reputação positiva ou negativa. As ações isoladas causam apenas algumas mudanças menos significantes no roteiro do jogo, que independentemente segue para o final.

O desfecho é afetado pela reputação do personagem principal. No caso, duas opções de finais existem, mas ambas deixam a situação em aberto. Ou seja, *inFAMOUS*, apresenta multiplos caminhos, multiplos finais e também final em aberto. Por outro lado, o enredo e toda a história deste jogo não é tão profunda quanto a do RPG analisado anteriormente, mas certamente mostra características importantes das narrativas nos jogos.

Provavelmente, casos reais são a melhor forma de se ver como uma narrativa é realmente implementada num jogo. Ambos os casos analisados obtiveram sucesso no mercado. É claro que não só pela narrativa, mas pelo jogo como um todo, mesmo assim percebe-se que houve um investimento em escrita.

3. Tecnologias para aplicações em dispositivos móveis

Dispositivos móveis são plataformas de jogos em ascensão. Segundo Crook (2009) pesquisas de mercado indicam um grande crescimento no mercado de jogos *mobile* até 2014. De forma similar, Killen (2011) indica um crescimento dramático no número de jogos, e aplicações em geral, sendo vendidas para este tipo de hardware. Essa presença crescente dos portáteis na vida das pessoas é consequência de uma natural evolução da tecnologia.

O desenvolvimento de aplicações para dispositivos móveis, bem como o desenvolvimento de jogos são assuntos bastante amplos. Os portáteis possuem muitas peculiaridades quanto a poder de processamento e especialmente interface. Do outro lado, os jogos apresentam uma complexidade de projeto e desenvolvimento relacionados a adequada utilização das mídias, como a mecânica do jogo (técnicas e codificação), imagens (cenários e personagens) e áudio (composição musical que traz amplia sensação de imersão e emoção).

Neste capítulo serão identificados alguns dos conceitos mais importantes para o desenvolvimento de jogos em plataformas móveis. Também serão analisadas algumas ferramentas de desenvolvimento que facilitam as tarefas de construção de um jogo.

3.1 Dispositivos móveis

Existem muitos tipos de dispositivos móveis. Celulares, *smartphones*, *tablets* e consoles portáteis são alguns. Todos eles possuem algumas características em comum, como telas pequenas, e baixo poder de processamento, como é apresentado no trecho a seguir:

> "Uma restrição importante do *design* para tecnologia móvel é o espaço limitado de tela ou a sua ausência. Outras características tecnológicas significativas incluem a duração da carga da bateria e pode haver limitações de armazenagem, memória e capacidade de comunicação" (BENYON, 2011).

Fica claro que os *smartphones*, e os portáteis em geral, são ainda muito limitados pela tecnologia. Isso força os desenvolvedores a tomar um direcionamento

diferente quanto ao tipo de produto que se destina a estes aparelhos. A forma como aplicações, *websites* e os próprios jogos são criados para uma plataforma de mesa é muito diferente da maneira utilizada nos portáteis.

O fator de *design* que se torna mais evidente para as aplicações em dispositivos móveis é a interface reduzida. Nielsen (2011, tradução do autor) afirma "para se ter sucesso em uma aplicação ou *site* para *mobile*, a diretriz óbvia é projetar para uma tela pequena". A tela de um portátil, por sua natureza diminuída, impede que se tenha muita informação disponível sem a necessidade de acesso a menus, e torna mais difícil desenvolver atividades como leitura. Considera-se também que os objetos da interface com os quais o usuário pode interagir devem ser grandes, para permitirem acionamento através de toque. Interfaces com botões ou menus pequenos podem prejudicar severamente a navegação e a experiência do usuário em geral, visto que são elementos de interface idealizados para ativação através de mouse. A complexidade da aplicação pode aumentar muito, dependendo da quantidade de características que forem implementadas. Nielsen (2011, tradução do autor), novamente, ilustra "quando você tem uma interface menor, você deve limitar a quantidade de recursos baseado naqueles que são mais importantes para o portátil".

O segundo principal limitador das aplicações para dispositivos móveis é a falta de recursos de processador e memória. Segundo Raento (2009, tradução do autor) "*smartphones* podem ser comparados a computadores pessoais da década de 90 em memória, disco rígido, capacidade de processamento e conexão com internet". Isso mostra que os portáteis ainda carecem de recursos. Mesmo assim, a criação de *softwares* de qualidade para os portáteis é uma realidade. Possivelmente isso se dê pelo fato de que os aparelhos móveis de hoje em dia possuem propósito diferente dos tais computadores mais antigos.

Um terceiro fator que aumenta a complexidade de criação de aplicações para dispositivos móveis é a variedade de aparelhos. As empresas desenvolvem aparelhos com sistemas operacionais, tamanhos de tela, botões e tecnologias diferentes. Benyon (2011) adiciona "Diferentes dispositivos têm diferentes formas de conectividade como *Bluetooth*, Wi-Fi, GPRS (Serviço de Rádio de Pacote Geral) e 3G". Fica a cargo do desenvolvedor a definição de qual plataforma irá produzir suas

aplicações. Porém, nos últimos anos, a própria evolução da plataforma móvel trouxe algumas padronizações, especialmente dentro de cada empresa. Apple fornece documentos que ajudam o desenvolvedor a produzir o seu software indicando melhores técnicas e, inclusive, princípios de interface. Esta forma está sendo seguida por outras empresas ligadas ao desenvolvimento de aplicações *mobile*.

3.2 Desenvolvimento de jogos

Criar jogos é uma ideia que fascina muitas pessoas. Talvez por envolver tantos tipos de trabalhos artísticos como ilustração e música, ou talvez por simplesmente ser lúdico. Porém, mesmo sendo interessante é uma tarefa de alta complexidade. McShaffry (2009, tradução do autor) afirma “o trabalho é incrivelmente difícil e pode te deixar completamente louco. As ferramentas e sistemas que você utiliza mudam mais do que deveriam. Alguns dias você remove mais código do que escreve”. Esta dificuldade existe pois é necessário criar muitas partes de *software* que necessitam de conhecimento de ciência da computação como gráficos, áudio e redes (Strougo e Wenderlich, 2011). Por outro lado, jogos são, essencialmente, programas como quaisquer outros e, portanto possuem recursos que podem ser utilizados para facilitar o desenvolvimento.

As principais características que precisam ser desenvolvidas em jogos, normalmente incluem gráficos, física e áudio. Cada um destes itens possui uma certa complexidade e, devido a isso, eventualmente, foram criados algoritmos e bibliotecas para auxiliar o desenvolvimento com cada uma destas partes.

Para criação de gráficos em jogos existem algumas ferramentas bastante populares como *OpenGL* (*Open Graphics Library*) e *Direct3D*. Ambas são capazes de criar mundos virtuais em três dimensões, e, devido a sua maturidade, proporcionam resultados de qualidade. Wright Junior, Lipchak e Haemel (2007, tradução do autor) descrevem *OpenGL* como “esta poderosa API (interface de programação de aplicativos) para criar impressionantes visualizações, jogos e outros gráficos de todos os tipos”. A parte gráfica tem um impacto forte na qualidade final do jogo, portanto sempre procura-se utilizar as melhores ferramentas para recursos visuais.

Para tratar a física dos jogos existem os chamados motores de física. Catto (2011) descreve *Box2D* como “uma biblioteca de simulação de corpos rígidos em 2D [...] usada para mover objetos de uma maneira natural e tornar o mundo do jogo mais interativo”. Esta descrição está de acordo com qualquer outro motor de física, inclusive os 3D. Muitos jogos dependem de uma simulação de qualidade e estes algoritmos tornam muito mais eficiente o desenvolvimento.

Seguindo com as interfaces de programação, no quesito áudio também existem APIs. Uma das mais famosas é o FMod, descrito como “um revolucionário motor de áudio para desenvolvedores de jogos, desenvolvedores multimídia, *designers* de som, músicos e engenheiros de áudio” (FMod, 2011, tradução do autor). Assim como gráficos e física, as interfaces dedicadas a áudio auxiliam os desenvolvedores a criar um produto de qualidade, sem necessitar desenvolver tudo a partir do zero.

É importante apontar que estas APIs são grandes trabalhos que muitas vezes levaram anos para chegar a um estado de maturidade. Estes trabalhos geraram resultados e melhorias nas condições de desenvolvimento de jogos, até que surgiram os SDKs (*kits* de desenvolvimento de software). “Um SDK é um conjunto de ferramentas projetadas para ajudar no desenvolvimento de uma tecnologia específica” (Novak, 2010). Encontra-se com frequência SDKs que unam as APIs gráfica, física e de áudio em um único produto. Muitas vezes os SDKs vão além disso, incluindo APIs dedicadas a comunicação e até mesmo inteligência artificial. De fato, os *kits* de desenvolvimento são muito utilizados na criação de jogos, porém, além dos SDKs, estão os motores de jogo.

A definição do que é um motor de jogo é bastante ambígua. Ele possui mais recursos do que uma API e um SDK, porém não é o jogo completo. Zerbst e Düvel (2004, tradução do autor) definem um motor como “uma parte do projeto que impulsiona certas funcionalidades do programa”. Thorn (2011, tradução do autor) adiciona ao conceito que “um jogo depende de seu motor a ponto de que sem ele o jogo não poderia ser desenvolvido ou jogado. Porém, isso não significa que um jogo é a mesma coisa que o motor. [...] o motor é o coração do jogo”. As definições não especificam os limites do motor de jogo dentro de um projeto, como fica explícito, também no texto que segue:

> "Há uma linha tênue entre um jogo e seu motor. Alguns motores possuem uma distinção razoavelmente clara, enquanto outros nem sequer tentam separar os dois. Em um jogo o código de renderização pode 'saber' especificamente como desenhar um *orc*[2]. Em outro jogo, o motor de renderização pode prover materiais e sombreamentos genéricos, e configurar os *orcs* em dados. Nenhum estúdio faz uma separação clara de jogo e motor, mas isso é perfeitamente entendível, considerando que as definições destes dois componentes muitas vezes se transformam enquanto o *design* do jogo se solidifica" (Gregory, 2009, tradução do autor).

Normalmente um motor é focado em um gênero de jogo como, por exemplo, a *Source Engine* desenvolvida pela Valve que é um motor dedicado a jogos de tiro em primeira pessoa. Segundo Rollings e Morris (2004) uma das maiores dificuldades de se fazer um motor de jogo é saber até que ponto levá-lo. Com um nível de especificidade baixo força os desenvolvedores a escreverem muito código, porém com excesso de especialização foca demais em apenas um gênero.

Embora existam estas questões teóricas os motores de jogo ajudam muito os desenvolvedores a produzirem grandes jogos.

3.3 Desenvolvimento de jogos em dispositivos móveis

Existem ferramentas de desenvolvimento para jogos dedicadas a dispositivos móveis. Isto inclui APIs miniaturizadas e, até mesmo, motores de jogo. Como este projeto é dedicado a esta área, serão apresentados algumas peculiaridades deste tipo de desenvolvimento.

Algumas bibliotecas crescem com o tempo e se tornam imensas. Muitas vezes são criadas novas funções que permitem que sejam feitas as mesmas ações que já eram atendidas antes, ou para melhorar a eficiência, ou para adaptar-se a um novo padrão de código. Mas fica claro que isso torna mais difícil a adaptação deste tipo de interface de desenvolvimento para plataformas móveis, visto que estas possuem menos recursos. Porém, projetos são criados para tornar possível a adaptação destas APIs para plataformas de menor desempenho. O exemplo perfeito para isso é *OpenGL ES* (*OpenGL for Embeded Systems*). Wright Junior, Lipchak e

[2] Orc é uma criatura humanóide com a pele em tom de verde.

Haemel (2007) o descrevem como “um conjunto de APIs destinado a uso em ambientes embarcados onde tradicionalmente os recursos são mais limitados.” Os autores ainda acrescentam que para criação desta interface de programação resumida foram tirados muitos métodos que faziam a mesma coisa, desinchando, assim, o pacote. Atualmente *OpenGL ES* é suportada na maior parte dos aparelhos celulares e *tablets*, e é, portanto, um padrão do mercado de video games.

OpenGL ES se diferencia do OpenGL padrão em dois aspectos principais. Primeiro não há suporte para as funções *glBegin* e *glEnd*[3], ao invés disso o programador deve utilizar vetores de vértices. Segundo, funções que funcionavam exclusivamente com cálculos em ponto flutuante e dupla precisão foram substituídos por cálculos de ponto fixo.

Tratando-se dos motores de jogo existem alguns dedicados unicamente a plataformas móveis, enquanto outros se adaptaram pelo mesmo processo de redução que outras APIs passaram. Como tirar proveito destes projetos depende do propósito do desenvolvedor.

3.4 Análise de Motores de Jogo

Para desenvolvimento do projeto proposto neste trabalho utilizar-se-á um motor de jogo. Existem algumas opções dedicadas a desenvolvimento em plataformas móveis. Foi feita uma comparação entre SDL, Cocos2D iPhone e *Unity3D*. Todos possuem muitas diferenças ao nível de desenvolvedor quando comparados aos outros, mas também todos possuem bastante destaque entre programadores e podem ser considerados boas opções, cada um em seu caso específico.

O primeiro analisado foi o SDL (*Simple Directmedia Layer*). Ele é descrito como “uma biblioteca multimídia e multiplataforma, que oferece acesso a áudio, teclado, mouse, controles, aceleração gráfica 3D e *buffer* de vídeo 2D.” (SDL, 2011, tradução do autor). Observando-se os conceitos de motor de jogo, talvez seja

---

[3] *glBegin* e *glEnd* são funções do *OpenGL* que identificam o início e o fim do processo de desenho de primitivas na tela.

possível dizer que o SDL não se enquadre na definição. Porém, esta biblioteca vem sendo utilizada em inúmeros jogos há muitos anos.

SDL, cuja estrutura é apresentada na Figura 2, tem um padrão de desenvolvimento simples. É uma biblioteca que simplifica o acesso a baixo nível, e, facilita o processo dos itens mais básicos do desenvolvimento de jogos, como a utilização de imagens ou de audio. Possui algumas extensões que são comumente utilizadas nos projetos deste motor. As extensões mais utilizadas são:

SDL_image: para tratamento de imagens em outros formatos além do .bmp, suportado pelo SDL padrão;

SDL_ttf: para utilização de *True Type Fonts*, ou, simplesmente para adição de texto na tela;

SDL_mixer: para tratamento de audio em geral.

A biblioteca foi desenvolvida em C, portanto o acesso a todos os recursos segue o padrão procedural da linguagem. O maior impacto é que no lugar de classes há *structs* que guardam as informações sobre o contexto do jogo.

Outro ponto importante de SDL é que a ferramenta oferece recursos gráficos através de *OpenGL*. Isso significa que, por baixo do SDL, ainda há uma API muito boa para tratamento de imagens, o que torna SDL mais eficiente e confiável.

O problema de se usar SDL é a falta de uma estrutura avançada de desenvolvimento de jogos. O que esta ferramenta fornece é apenas o acesso bruto aos recursos básicos como mouse e teclado. O desenvolvedor deve construir o *software* desde os níveis mais baixos, incluindo itens como estrutura de animação e gerenciamento de cenas. Torna-se mais demorado produzir um jogo inteiro tendo apenas SDL como base. Por ouro lado, esta biblioteca estaria num nível perfeito para se criar um motor de jogo pois é muito eficiente em termos de processamento e não causa nenhum tipo de limitação ao desenvolvedor.

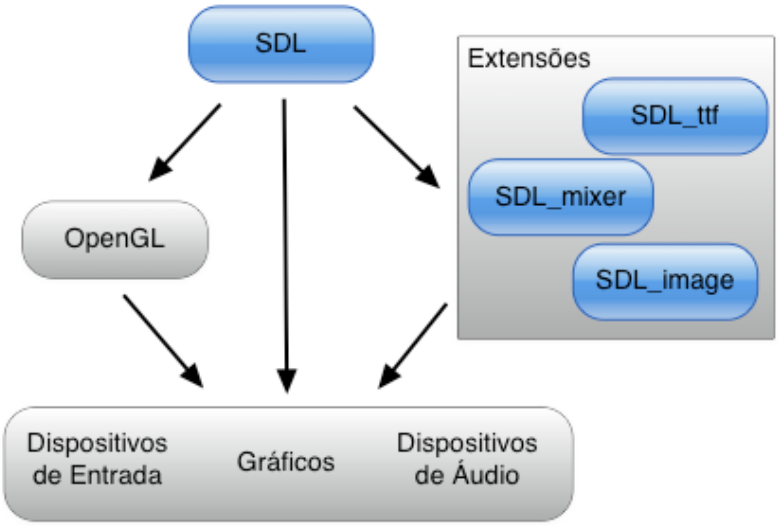


Figura 2 - Estrutura do SDL.

O segundo motor analisado foi o Cocos2D iPhone. É descrito como "um *framework* para criação de jogos em 2D, demos e outros aplicativos gráficos/ interativos." (Cocos2D, 2011, tradução do autor). Cocos surgiu como uma ferramenta em *Python*, mas foi portada para *Objective-C* para atender as plataformas da Apple. Seu foco é unicamente em iOS e Mac. A Figura 3 ilustra a estrutura básica de como Cocos2D funciona.

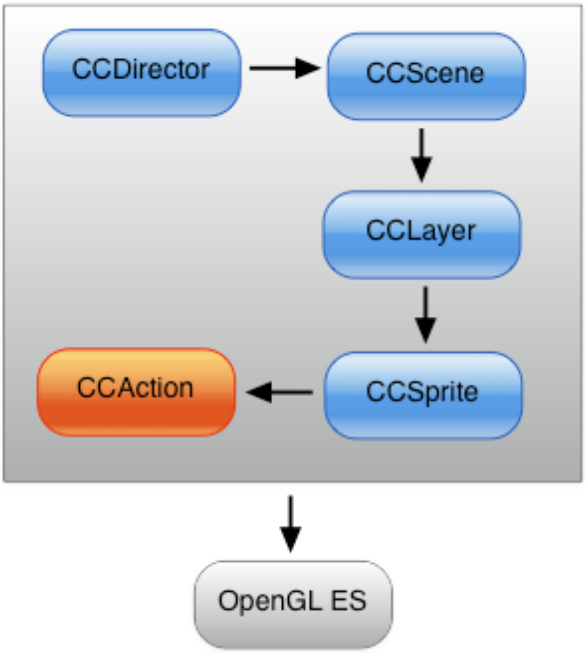


Figura 3 - Estrutura do Cocos2D.

A nomenclatura das classes é um padrão da linguagem *Objective-C* proposto pela Apple. “Utiliza-se prefixos para evitar conflitos de símbolos desenvolvidos por terceiros ou pela Apple” (Apple, 2011). Portanto, dentro do *framework* Cocos2D as classes possuem nome iniciado por “CC”.

Cocos2D foi desenvolvido para funcionar sobre OpenGL ES, ou seja, já foi projetado para ser utilizado em desenvolvimento para dispositivos móveis. Sua estrutura foi feita baseando-se em 4 itens principais: Diretor, cenas, camadas e *sprites*. Cada um destes é representado por uma classe no *framework*.

Começando do item mais baixo, um ***sprite*** representa um objeto da interface, seja personagem, objeto, menu, ou praticamente qualquer elemento visual. Estes *sprites* compõem as camadas. Uma **camada** é, literalmente, uma camada na interface do jogo. Por exemplo, uma camada pode mostrar um plano de fundo, enquanto outra camada apresenta o personagem sobre uma plataforma. A união de várias camadas forma uma cena. O conceito de **cena** é feito para representar cada estado do jogo. Uma cena pode identificar o menu do jogo, enquanto outra cena representa a estrutura *in game* do jogo. O **diretor** é quem vai controlar as cenas e suas transições. Cada um desses itens é representado por uma classe independente que são utilizadas em conjunto através de composição. As classes no *framework* são chamadas de *CCSprite, CCLayer, CCScene,* e *CCDirector*.

Cocos2D iPhone, possui uma biblioteca com conteúdo o suficiente para se construir rapidamente jogos e é uma ótima ferramenta especialmente se tratando de plataformas móveis.

Unity3D foi o terceiro a ser analisado. O motor é definido como “uma ferramenta de desenvolvimento de jogos que foi feita para permitir que você se foque em criar jogos incríveis.” (*Unity3D*, 2011, tradução do autor). Esta ferramenta é a que se assemelha mais com os conceitos de motor de jogo, pois inclui não somente uma grandiosa API com muitos recursos, formulada para desenvolver jogos, mas também uma ferramenta que permite construir o jogo graficamente. Além disso, *Unity3D* permite que o desenvolvedor crie jogos para plataformas móveis da mesma maneira que criaria para plataformas desktop.

Este motor possui um conceito de cena semelhante ao do Cocos2D. “Cenas contém os objetos do seu jogo. Elas podem ser usadas para criar um menu, níveis do jogo ou qualquer outra coisa.” (Unity3D, 2011, tradução do autor). O desenvolvedor utiliza o ambiente para inserir objetos, editar suas propriedades e criar as cenas de jogo.

Esta ferramenta, além de prática para o desenvolvedor ainda traz uma grande quantidade de recursos prontos, incluindo *scripts* de controle de mouse e movimentação de personagens. Este motor torna fácil o desenvolvimento, até mesmo para quem não conhece programação. Os únicos detalhes negativos do *Unity3D* são, primeiro, que ele é exclusivamente 3D. É possível criar jogos em 2D, mas o motor carrega e utiliza todo o sistema em 3D, o que pode tornar o produto final ineficiente. E o segundo ponto negativo é o preço da licença para desenvolvimento dedicado a iOS e *Android* que custa mais do que muitos desenvolvedores desejam pagar.

A Tabela 1 abaixo mostra um resumo dos conceitos comparados entre as três plataformas.

Tabela 1 - Comparação entre motores de jogo.

Nome	**SDL**	**Cocos 2D iPhone**	**Unity3D**
Linguagem	C	Objective-C	Boo, C#, Javascript
Custo	Gratuito	Gratuito	$400 +
Exemplos notáveis	World of Goo, Aquaria, Braid	Vários jogos menores para iPhone	ShadowGun, Gears, Battlestar Galactica
Controle de baixo nível	Muito alto	Baixo	Baixíssimo
Plataformas Padrão	Windows, Linux e Mac	iOS, Mac	Windows, Mac, iOS, Android e Browsers

Recursos prontos	Básico	Quantidade razoável	Grande quantidade
Maturidade	Muitos anos	Recente	Alguns anos

3.5 Testes de implementação com motores de jogo

Para uma comparação um pouco mais profunda, não apenas teórica, foi feito um teste de implementação. O objetivo era encontrar as dificuldades de instalação e utilização, bem como descobrir a velocidade de desenvolvimento. O teste consistiu em criar um projeto utilizando cada uma das tecnologias, inserir um objeto na tela e animá-lo.

A instalação do SDL possui uma complexidade média. Utilizando-se o ambiente de desenvolvimento *Xcode*, é necessário fazer ajustes para o funcionamento de maneira ideal, uma vez que não se tem um projeto modelo pronto. Primeiro cria-se um novo projeto e, nele, adiciona-se as bibliotecas do motor. Também é importante adicionar estas mesmas bibliotecas à uma fase de cópia no processo de compilação (Figura 4). Com isso, as bibliotecas são copiadas para o executável final e permitem a execução do aplicativo sem que o usuário final precise instalar o SDL.

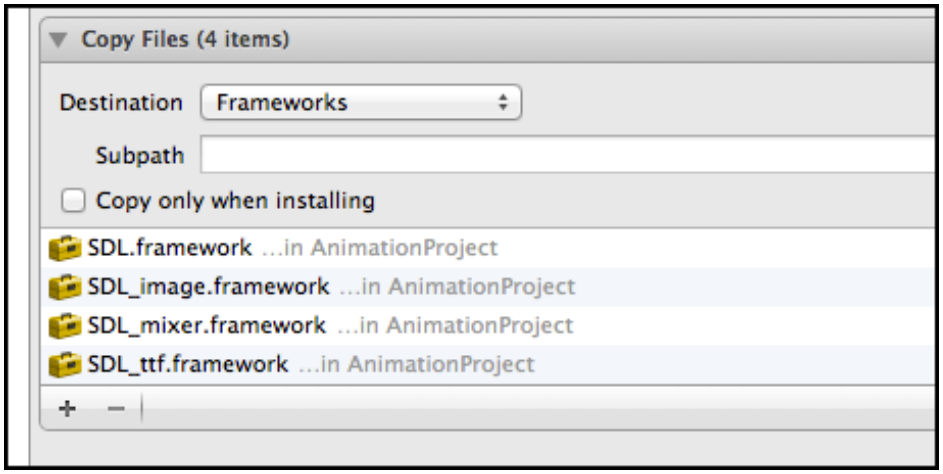


Figura 4 - Diretiva de cópia de bibliotecas para executável.

Para execução do teste foi produzido um experimento utilizando a linguagem de programação *Objective-C*. Esta é um superconjunto de C, portanto compatível

com SDL. O Código 1 foi feito para ser o mais simples possível. A função animar retorna um número, referente ao quadro atual da animação, de acordo com o tempo de execução do aplicativo. Na função *main* é criado um vetor de quadros (*SDL_Surface*), representando a animação inteira. Então há um laço infinito que limpa a tela e desenha o quadro atual da animação.

```
000.  - (int)animar {
001.      if (tempoAntigo + taxaDeQuadros < SDL_GetTicks()) {
002.          tempoAntigo = tempoAntigo + taxaDeQuadros;
003.          quadroAtual += 1;
004.          if (quadroAtual >= maximoDeQuadros - 1)
005.              quadroAtual = 0;
006.      }
007.      return quadroAtual;
008.  }
009.
010.  - (void)main {
011.      SDL_Surface *tela;
012.      SDL_Surface *quadros[4];
013.      SDL_Rect pos;
014.
015.      SDL_Init(SDL_INIT_VIDEO);
016.      tela = SDL_SetVideoMode(800, 600, 0, SDL_HWSURFACE);
017.
018.      quadros[0] = IMG_Load("img1.png");
019.      quadros[1] = IMG_Load("img2.png");
020.      quadros[2] = IMG_Load("img3.png");
021.      quadros[3] = IMG_Load("img4.png");
022.
023.      pos.x = 142;
024.      pos.y = 44;
025.
026.        while(YES) {
027.          SDL_FillRect(tela, NULL, 0xffffff);
028.          SDL_BlitSurface(quadros[[self animar]], NULL, tela, &pos);
029.          SDL_Flip(tela); //flush
030.      }
031.  }
```

Código 1 - Experimento em SDL.

O teste com Cocos2D ocorreu sem impedimentos. Como descrito na seção anterior, este motor possui alguns conceitos diferentes do SDL, portanto, o experimento tomou uma forma diferente. Porém, os passos são os mesmos. Primeiro carrega-se as imagens e depois itera-se sobre elas em um dado tempo. Cocos2D utiliza *Objective-C* como linguagem de programação. O Código 2 também segue o princípio da simplicidade. A classe *CCSpriteFrameCache* é responsável por carregar na memória uma imagem subdividida em imagens menores, cada uma representando um quadro da animação. Depois de carregada a imagem, cria-se um

vetor para conter cada imagem menor, e cria-se uma *CCAction*, responsável por controlar o funcionamento da animação. Ao iniciar a execução da aplicação, dispara-se a ação e a animação é executada pelo motor de jogo e apresentada na tela.

```
000.  CGSize tamanho = [[CCDirector sharedDirector] winSize];
001.  CGPoint posicao = ccp(tamanho.width / 2, tamanho.height / 2);
002.  CCSprite *quadro = nil;
003.  CCSprite *fundo = [CCSprite spriteWithFile:@"fundo.png"];
004.  fundo.position = posicao;
005.
006.  [[CCSpriteFrameCache sharedSpriteFrameCache]
007.                addSpriteFramesWithFile:@"quadros.plist"
008.                            textureFile:@"anim.png"];
009.
010.  NSMutableArray *frames = [NSMutableArray arrayWithCapacity:8];
011.  for (NSUInteger i = 0; i < 4; i++) {
012.      [frames addObject:[[CCSpriteFrameCache sharedSpriteFrameCache]
013.      spriteFrameByName:[NSString stringWithFormat:@"anim%d", i]]];
014.  }
015.
016.  quadro = [CCSprite spriteWithSpriteFrameName:@"anim0"];
017.  CCAnimation *animacao = [CCAnimation animationWithFrames:frames
018.                                                     delay:0.12];
019.  CCAction *acao = [CCRepeatForever actionWithAction:
020.                          [CCAnimate actionWithAnimation:animacao
021.                                    restoreOriginalFrame:NO]];
022.
023.  quadro.position = posicao;
024.  [quadro runAction:acao];
025.
026.  [self addChild:fundo z:0];
027.  [self addChild:quadro z:1];
```

Código 2 - Experimento em Cocos2D.

Utilizando-se *Unity3D* foi feito um experimento diferente dos outros. Para se construir a animação não foi necessário nenhuma linha de código sequer. Primeiro foi inserido um cubo 3D e uma fonte de luz no ambiente. Em seguida, foi criada a animação do cubo através de uma interface de edição de propriedades (Figura 5). Nesta interface o desenvolvedor vai indicar as transformações de propriedades do objeto e o tempo que elas demoram para acontecer. Após estes passos verifica-se na tabela de propriedades do cubo que foi adicionada uma animação a este (Figura 6).

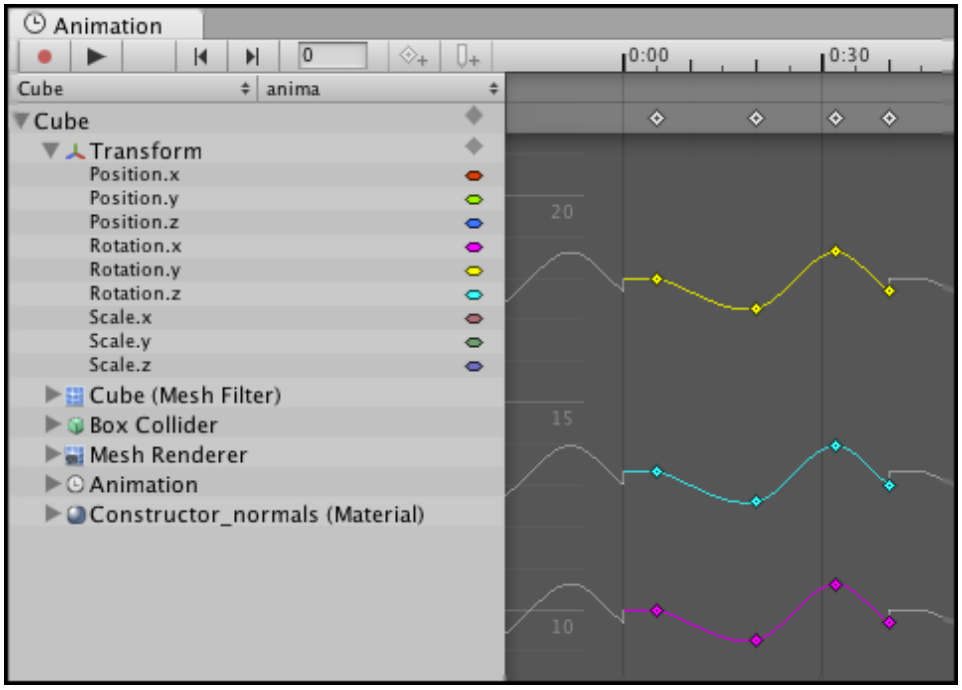


Figura 5 - Criação de um efeito de animação com Unity3D.

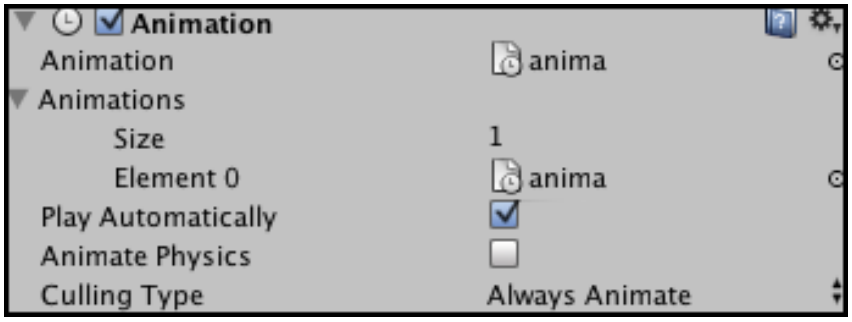


Figura 6 - Aplicando a animação a um objeto.

A análise dos motores de jogo descrita nesta seção apresenta alguns detalhes sobre o desenvolvimento de jogos especialmente dedicado a plataformas móveis. Para este trabalho, após verificadas as vantagens e desvantagens, foi decidido a utilização do motor Cocos2D iPhone. A princípio, o desenvolvimento com este motor deve ser mais rápido do que com a utilização de SDL, e considerando-se o preço do *Unity3D*, a melhor opção parece ser, mesmo, o Cocos2D.

4. Definição do jogo

A engenharia de software é uma caixa de ferramentas de onde os desenvolvedores podem pegar as ferramentas que necessitam (Flynt e Salem, 2005). A proposta deste trabalho é a construção de um jogo que permita ao jogador a criação de uma pequena narrativa. Partindo deste princípio foi idealizado o jogo **Story Teller** com o objetivo de apresentar uma jogabilidade simples e capacidade de criação de conteúdo. Para atender necessidades de estruturação do **Story Teller** serão apresentados alguns modelos de como o jogo deverá ser construído, e serão utilizados, também, diagramas de caso de uso e de classes conceituais, segundo os padrões UML (Linguagem Unificada de Modelagem).

A criação de um jogo, assim como de outros *softwares*, é composta por várias fases. Segundo Perucia (2005), o desenvolvimento de um jogo segue as etapas de *game design*, documentação, desenvolvimento e testes.

- *Game design*: é a fase de conceituação do software, onde se identifica como o jogador irá interagir com o jogo. Normalmente os envolvidos na produção do jogo se reúnem e levantam ideias e possibilidades para o futuro jogo.
- Documentação: desenvolve-se um documento de jogo[4] com a especificação dos itens idealizados durante a fase de *design*.
- Desenvolvimento: neste ponto é criado o conteúdo do jogo, juntamente com as possíveis fases e modos de jogo.
- Testes: nesta fase são aplicados testes de uso do software, para identificar possíveis erros e corrigi-los.

Um documento de jogo possui bastante importância na fase de concepção do jogo (Flynt e Salem 2005; Novak 2010). Este documento especifica o funcionamento do jogo e permite um melhor entendimento geral do que é esperado no produto final. Sua finalidade é explicada:

> "A documentação atende a duas finalidades: garantir que os membros da equipe compreendam suas respectivas funções no processo de desenvolvimento e convencer outras empresas a desenvolver, financiar a

[4] Em inglês é chamado de *GameDoc* ou *Design Document*

> produção ou ajudar de outras maneiras a transformar o *game* em realidade" (Novak, 2010).

Este tipo de documento não possui uma forma padronizada, mas possui algumas diretivas dos elementos mais importantes para descrição do jogo, como:

- Descrição: descrição do que é o jogo e qual seu objetivo.
- Motivação: elementos que fazem com que o jogador sinta vontade de jogar o jogo.
- Diferencial: elementos únicos do jogo que o diferenciam dos concorrentes.
- Gênero: indicação a qual classe de jogos o determinado jogo pertence.
- Público alvo: indicação de quem serão os prováveis maiores interessados no jogo.
- Forma de jogo: maneiras que o jogador terá para jogar o jogo.
- Diretivas de arte: indicação de como será a arte do jogo.
- Arte conceitual: exemplo de arte do jogo.
- Interface do jogo: exemplo de interface (protótipo).

Para o projeto **Story Teller**, o documento de jogo estará descrito na seção 4.1 deste trabalho.

Outro artefato de utilidade para desenvolvimento de software são os protótipos. Serão produzidos alguns exemplos de como é esperada a interface do software na aplicação final.

4.1 Conceito e Design - Documento de Jogo

4.1.1 Descrição

Story Teller é um jogo de criação de conteúdo. Esta criação será feita visualmente, se aproveitando dos recursos oferecidos pela plataforma de destino. O jogo **Story Teller** se enquadra no gênero RPG, e será desenvolvido para plataforma iOS, podendo ser expandido posteriormente para Android.

4.1.2 Motivação

Considera-se que as pessoas possuem um certo desejo de contar suas histórias e o **Story Teller** pretende tornar isso possível de uma maneira agradável. A criação de conteúdo também deve ser um motivador pois é um atrativo que tornou famosos alguns jogos como Little Big Planet (2008) e Minercraft (2009). **Story Teller**, no entanto, será mais focado em jogadores casuais e, portanto, deve ser uma experiência bem menos complexa do que os títulos citados.

4.1.3 Diferencial

A criação de conteúdo num smartphone deve ser o maior diferencial que o Story Teller pode apresentar. É uma plataforma pouco comum para criação de conteúdo, porém, se este for apresentado de forma fácil de usar pode se tornar interessante.

4.1.4 Gênero

O **Story Teller** é jogo do tipo RPG.

4.1.5 Público Alvo

Este jogo foi desenvolvido para usuários jovens, a partir de 10 anos, que não utilizam frequentemente videogames, e com familiaridade com o uso de dispositivos móveis.

4.1.6 Forma de Jogo

Quando um jogador estiver jogando este jogo ele deve ter a sensação de liberdade para que possa dar vida e expressar suas ideias. Para isso foi estruturada uma maneira de deixá-lo o mais confortável possível. O jogo se divide em dois modos principais:

- **Edição**: permite que o jogador crie histórias com cenários, personagens e eventos para posterior visualização.
- **Apresentação**: permite que o jogador controle um personagem e interaja com as histórias criadas.

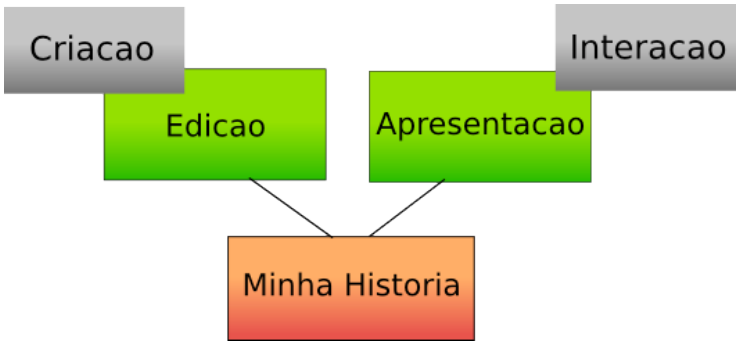


Figura 7 - Estrutura lógica do jogo.

A criação de conteúdo está estruturada em torno da criação de eventos que servem de roteiro e conduzem a história. O jogador criará sequências de eventos que darão forma a história do jogo e, posteriormente, ele poderá visualizar e interagir com o resultado criado.

4.1.7 Diretivas de Arte

O estilo da arte dos personagens do **Story Teller** é baseado em manga[5], um estilo popular em vários jogos incluindo a série Disgaea (2003). O estilo geral do software procura uma aparência de papel envelhecido, para atrair a ideia de histórias em meio impresso (livros). A combinação dos dois tipos de trabalho gera um estilo único que, embora simples, dá uma interface característica para o jogo. O Anexo A apresenta o material artístico produzido pela acadêmica de Bacharelado em Tecnologias Digitais, Anne Lorandi Pagno, para este software.

4.1.8 Arte Conceitual

Figura 8 - Estilo de arte Manga e fundo de papel envelhecido

[5] Estilo de arte de histórias em quadrinhos japonesas.

4.1.9 Interface do Jogo

O fluxo de utilização deste *software* é representado pela Figura 8. Inicialmente o usuário terá um menu onde selecionará se irá criar uma nova história ou abrirá uma previamente salva, sendo redirecionado para sua respectiva tela. Após a seleção, ou criação, da história ele poderá entrar em modo de edição, adicionando conteúdo à história, ou partir para visualização da mesma, onde irá interagir com os eventos que ele mesmo criou.

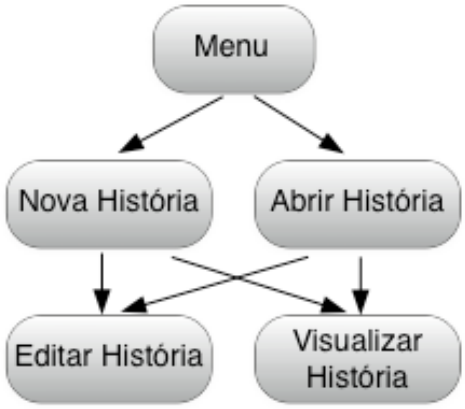


Figura 9 - Fluxo de utilização do jogo.

Considerando-se a plataforma de distribuição, a interface de *hardware* não irá conter botões físicos, apenas virtuais através da tela sensível ao toque. Este fator pode ser vantajoso pois permite que o desenvolvedor posicione todos os elementos da interface onde desejar, porém força a utilização de espaço de tela para botões e menus, em vez de outros elementos importantes para aplicação.

A interface de *software* seguirá os princípios de simplicidade, seguindo as recomendações de projeto propostas por Nielsen (2011). O modo de edição, porém, necessitará de mais opções de navegação do que o modo de apresentação, pois o jogador precisará tratar personagens, eventos e o cenário de forma individual. No modo de apresentação a interface terá menos funções, mas não perderá a interatividade.

Figura 10 - Exemplo de interface.

A Figura 9 apresenta alguns elementos de interface do modo de edição de história. Seguindo os padrões de interface discutidos, os botões de navegação são grandes, facilitando a utilização através da tela sensível ao toque. A interface pequena, infelizmente, não comporta muitos recursos de uma só vez. Assim, o projeto de interface inclui somente recursos essenciais para utilização do jogo o que, de certa forma, amplia a usabilidade do software e converge para a heurística de usabilidade que recomenda a omissão de informações irrelevantes em uma interface. Esta fragmentação do conteúdo força o usuário a navegar em diferentes menus e telas para editar as informações que necessita.

4.2 Modelagem do Software

Dada a contextualização do que é o jogo pode-se começar a definir diagramas UML para o *software*. O jogador possui algumas opções de utilização do

software. Em modo de edição ele cria, edita e transforma histórias, personagens e eventos, enquanto no modo de apresentação ele ativa eventos e executa ações.

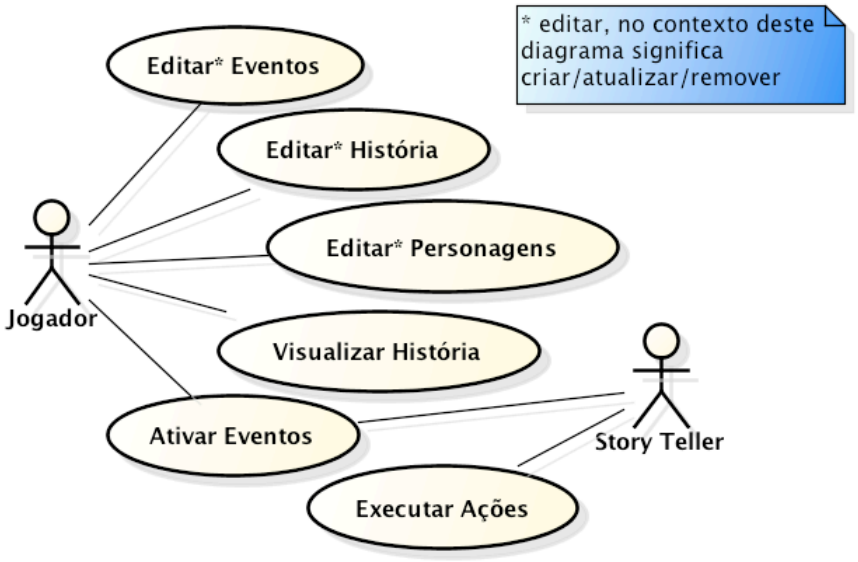


Figura 11 - Funcionalidades do sistema.

Este diagrama de caso de uso apresenta as situações de interação do jogador com o software. Os atores são jogador e o sistema (**Story Teller**). O jogador interage com toda a parte de edição e criação da história, bem como a parte de visualização. O sistema é um ator responsável por executar as ações disparadas pelo jogador e por outros eventos.

O desenvolvimento do **Story Teller** será feito em Cocos2D, e, portanto, seguirá os conceitos de diretor, cena, camada e *sprite* do motor de jogo. Em vista disso o jogo está dividido nas seguintes cenas:

- **Menu principal**: apresenta o menu principal com as opções de nova história e seleção de história.
- **Menu de Seleção de História**: apresenta uma lista de histórias salvas que o usuário possa abrir para editar ou visualizar.
- **Modo Edição de História**: apresenta a tela principal do jogo, que apresenta as opções de edição de história.

- **Modo de Apresentação de História**: apresenta a história selecionada para o jogador juntamente com a interface do jogo.

As cenas possuem uma grande importância no contexto do código, pois cada uma delas centraliza uma lógica de execução do software em diferentes partes. É justamente nas cenas que a lógica de cada parte do software tem início. Por causa disso as cenas foram separadas em classes (Figura 11).

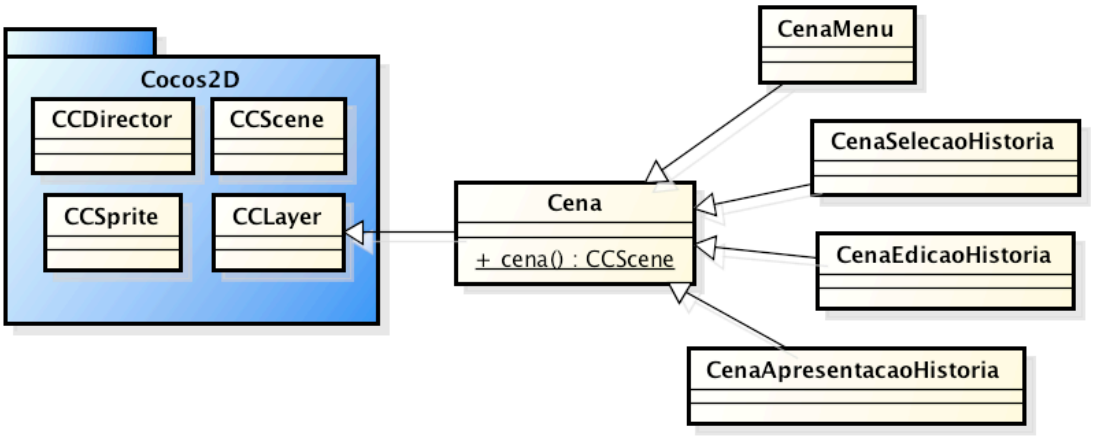


Figura 12 - Modelo conceitual referente às cenas.

Cada cena será uma generalização de uma classe Cena, que, por sua vez é uma generalização de *CCLayer*. Apesar de aparentar uma lógica errada, visto que Cena deveria herdar de *CCScene*, esta é uma prática comum em Cocos2D, pois, desta maneira, cada classe de cena, mesmo as mais simples, centralizam todo o código de desenho e atualização e, ao mesmo tempo, permitem que a própria classe da cena já possua conteúdo de interface. De fato, o diretor vai acessar as cenas através do método *cena()* definido na classe Cena. Este método é estático e deve retornar uma cena contendo a camada definida na própria classe. Subcamadas podem ser criadas separadamente para compor cada uma destas cenas. O Código 3 foi produzido para apresentar um exemplo genérico de uma classe de cena.

```
000.  #import "CenaExemplo.h"
001.
002.  @implementation CenaExemplo
003.
004.  + (CCScene*)cena {
005.      CCScene *cena = [CCScene node];
006.      CenaExemplo *camada = [CenaExemplo node];
007.      [cena addChild: camada];
008.      return cena;
009.  }
010.
011.  - (id)init {
012.      if ((self = [super init])) {
013.          CCLabelTTF *rotulo =
014.              [CCLabelTTF labelWithString:@"Ola Mundo"
015.                                  fontName:@"Marker Felt"
016.                                  fontSize:64];
017.          CGSize tamanho = [[CCDirector sharedDirector] winSize];
018.          rotulo.position = ccp(tamanho.width/2 ,tamanho.height/2);
019.          [self addChild: rotulo];
020.      }
021.      return self;
022.  }
023.
024.  - (void)dealloc {
025.      [super dealloc];
026.  }
027.
028.  @end
```

Código 3 - Implementação de classe de cena

O código apresenta o método “cena” com a inicialização de uma nova *CCScene*, seguido da criação de uma camada e subsequente inclusão da camada na cena. O método “*init*” possui função semelhante ao conceito de construtor, ou seja, é executado quando é instanciada a classe. Neste método é criado um rótulo e adicionado na camada. Este código é suficiente para apresentação do texto “Ola Mundo”.

Porém, naturalmente, somente as cenas não seriam o suficiente para se construir o jogo inteiro. Por isso foi criado um grupo de classes chamado de núcleo do jogo. Este grupo é formado por classes que representam os componentes da história.

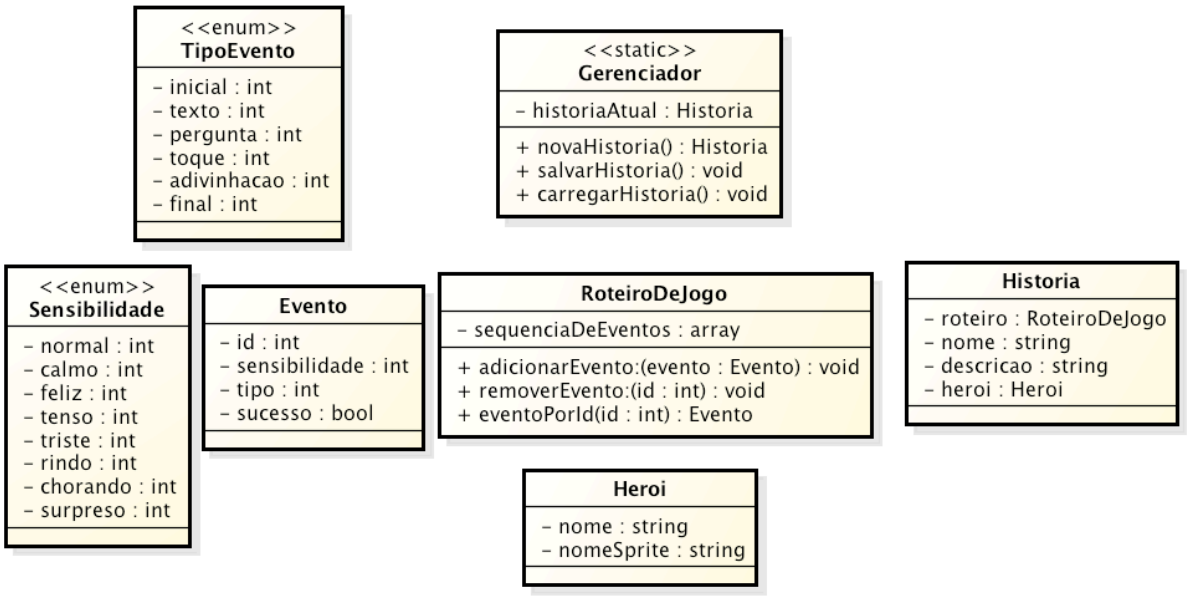


Figura 13 - Modelo conceitual referente ao núcleo do jogo

Estas classes servirão para construção e manutenção das informações da história.

- **Gerenciador**: responsável pela manutenção das histórias, carregando e salvando-as.
- **História**: centralizador de informações, contém atributos da história, referência ao roteiro e ao herói.
- **Herói**: guarda as informações do personagem principal da história.
- **Evento**: guarda as informações do evento que será interpretado durante a apresentação da história.
- **TipoEvento**: enumeração que identifica o tipo de cada evento.
- **Sensibilidade**: enumeração que representa a sensibilidade que cada evento irá conter.

O núcleo do jogo está estruturado em torno de eventos. Para explicar mais detalhadamente, um evento e as suas ações são elementos que o jogador adiciona ao roteiro com o intuito de dar dinâmica e interatividade a apresentação do jogo. Os eventos podem ser de diversos tipos, como por exemplo:

- **Inicial**: evento que representa o início da história. Serve como ponto de partida para todas as histórias e não pode ser removido.
- **Texto**: apresenta uma mensagem de texto para o jogador.
- **Pergunta**: apresenta uma mensagem de texto para o jogador com duas opções de resposta.
- **Toque**: mostra um "*mini-game*" onde ele deve tocar em botões que aparecem em posições aleatórias da tela.
- **Adivinhação**: mostra três botões no qual apenas um é o correto.
- **Final**: define o término da história. Após este evento não podem ser adicionados novos eventos.

A sensibilidade dos eventos representa a forma como cada evento será apresentado. Um evento com sensibilidade feliz conterá uma trilha sonora mais animada e cores que representem o estado de felicidade, enquanto outro evento com sensibilidade tensa apresentará trilha sonora com distorção e cores mais escuras. A seção 5.2 apresenta uma explicação mais detalhada da sensibilidade.

Os eventos serão configurados por código. Portanto, a adição de novos tipos de eventos e ações se dará somente através de recompilações do projeto.

Com o núcleo e as cenas, a estrutura geral do software está disposta, mas ainda é necessário mais um grupo de classes auxiliares com responsabilidades específicas.

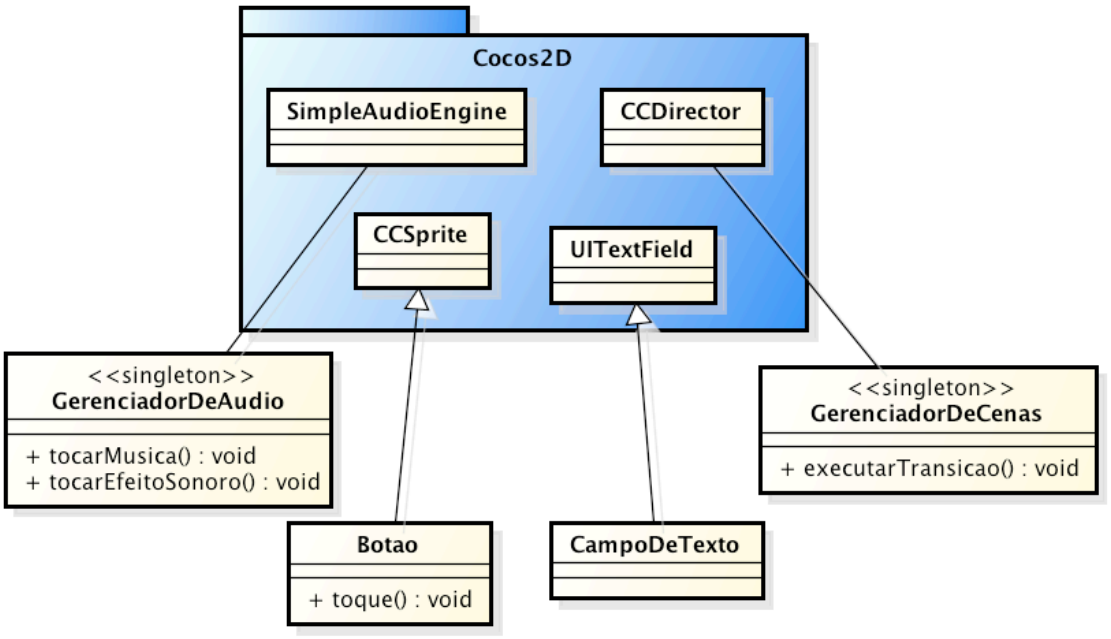


Figura 14 - Modelo conceitual referente às classes auxiliares.

Apesar de poder acessar o código do motor de jogo, foi decidido não fazer alterações nele, e sim, trabalhar apenas com extensões e classes intermediárias. Os gerenciadores existem por causa disso. Já as classes Botao e CampoDeTexto foram criadas para tratar os comandos de entrada do jogador, especialmente durante o modo de apresentação do jogo.

- **GerenciadorDeAudio**: responsável por gerenciar tanto a música quanto efeitos sonoros. Evita repetição de código de áudio. A classe segue o padrão singleton[6] e está acessível em qualquer parte do projeto.
- **GerenciadorDeCenas**: responsável por tratar as mudanças de cenas, incluindo transições ou efeitos possíveis, e, a exemplo da classe GerenciadorDeAudio, segue o padrão singleton enquanto acessível em qualquer parte do projeto.
- **Botao**: é uma generalização de *CCSprite* capaz de reagir a toques na tela. Essa funcionalidade dá características de botão a classe.
- **CampoDeTexto**: responsável por tratar o a entrada de texto dada pelo jogador.

[6] padrão de projeto que garante a existência de uma única instância do referido objeto.

Os diagramas apresentados permitem uma visão geral da estrutura do software. Com isso pode-se definir uma arquitetura geral do jogo **Story Teller**.

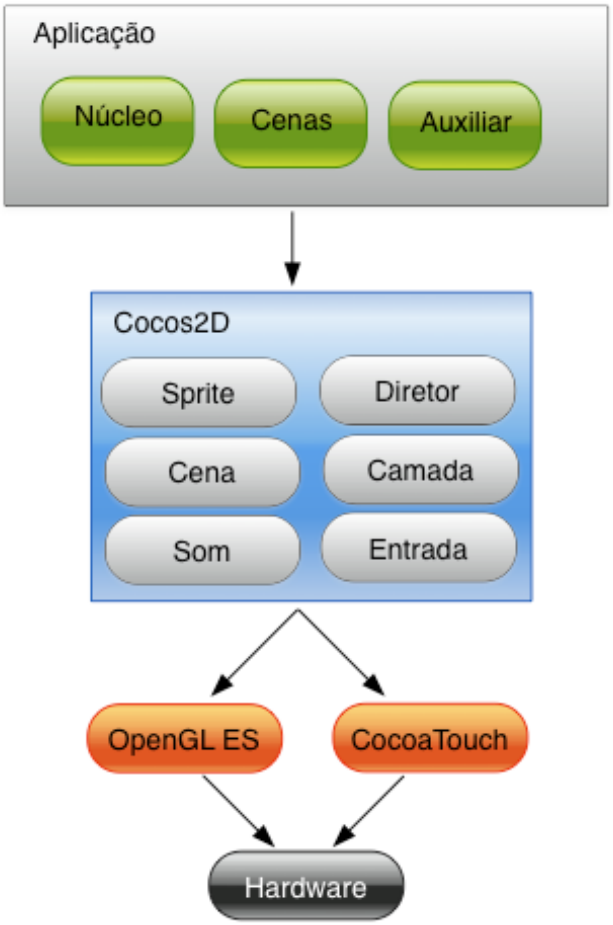


Figura 15 - Arquitetura geral do jogo.

Cada componente da arquitetura tem uma função específica. Porém, todos eles em conjunto tornam possível a criação do jogo **Story Teller**. O motor de jogo é dependente das estruturas de *OpenGL ES* e *CocoaTouch*[7], enquanto o código de alto nível, pertencente a aplicação, é dependente do motor. Pela união de todos os poderes é possível construir um jogo.

Ao abrir o jogo, o jogador irá se deparar com algo similar aos protótipos apresentados. Mesmo que as imagens não tenham alta fidelidade à versão final do software, os protótipos ajudam bastante a visualizar e entender o que está sendo projetado.

[7] API padrão de desenvolvimento para iOS.

Figura 16 - Tela de abertura à esquerda e tela de seleção à direita.

Figura 17 - Modo de apresentação à esquerda e modo de edição à direita.

Figura 18 - Edição de informações à esquerda e edição de evento à direita.

Os protótipos devem causar um melhor entendimento de como será utilizado o software final.

5. O software

Story Teller se trata de um software onde o usuário constrói e interage com histórias. Ele apresenta uma forma simples de construção e de visualização de roteiros. Se assemelha, um pouco, a jogos bastante antigos como o *Colossal Cave Adventure* (1976), mencionado no capítulo 1, que é uma aventura inteira em modo texto onde o jogador interage de formas mais primitivas com o software. O **Story Teller** é um jogo que busca explorar a criatividade de quem o joga. Para atingir tal objetivo utiliza-se de recursos para tornar a interatividade do jogador mais interessante.

5.1 Os eventos e a interatividade

Como citado na seção 4.2, o **Story Teller** utiliza-se de eventos para montar um roteiro. Os eventos são de vários tipos, sendo que alguns possuem mais de um caminho de saída, dando possibilidades para o criador da história montar caminhos alternativos e múltiplos finais. Além dos caminhos variados existe a possibilidade de *mini-games*, que são jogos mínimos que duram apenas alguns segundos, e podem ser inseridos dentro da história.

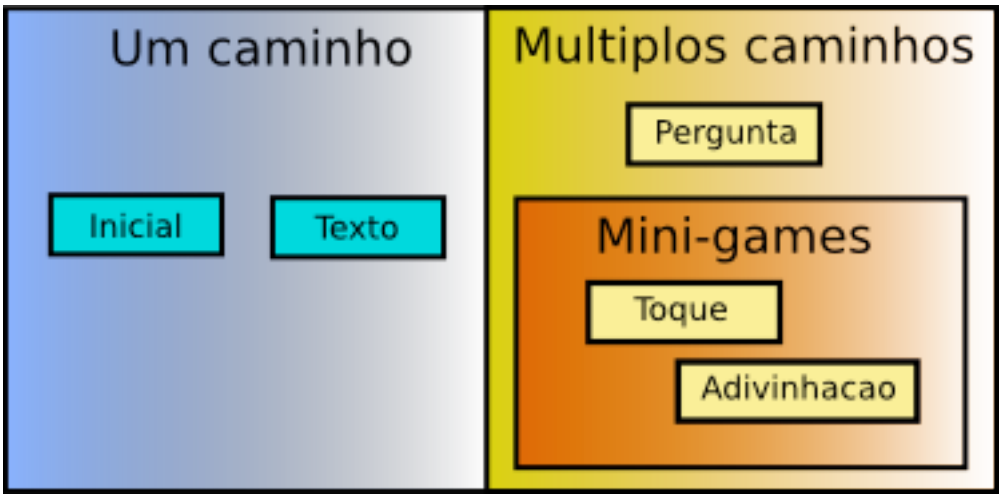


Figura 19 - Fluxo de tipos de eventos

Os eventos de caminho único apenas apresentam seu conteúdo, e em sequência permitem que o jogador continue assistindo o próximo evento. O evento inicial é único e não pode ser inserido pelo jogador. O evento texto é o tipo de evento

base para toda narrativa, uma vez que com ele é possível apresentar mensagens ao jogador.

Os eventos de múltiplos caminhos possuem dois caminhos de saída. Isso significa que quando o jogador chegar num ponto do roteiro onde há um evento de múltiplos caminhos, este acabará seguindo por um de dois lados possíveis. O evento “pergunta” apresenta um texto ao jogador, e este irá decidir entre duas possíveis respostas.

Os eventos *mini-game* são um caso especial pois eles não apresentam texto, apenas uma situação mais dinâmica ao jogo. O evento **toque** apresenta botões que aparecem em posições aleatórias da tela e cabe ao jogador acertar o maior número de cliques possível. Este evento foi inspirado nas Leis de Fitt (Preece, 2005) que define que a dificuldade de atingir um alvo é uma função da distância do alvo e do seu tamanho. O evento **adivinhação** apresenta três opções ao jogador, se este acertar segue por um caminho, senão ele segue pelo outro caminho. Os eventos aleatórios são comuns em jogos para causar uma dinâmica diferente a cada vez que se joga. Jogos como o de adivinhação permitem um fluxo que não depende do raciocínio ou habilidade do jogador, mas simplesmente da sorte. Pode-se, até mesmo, comparar com o rolar de dados. O resultado positivo ou negativo depende de fatores além do controle do usuário.

Figura 20 - Mini game de toque. Possui botões grandes.

Mesmo dentro dos *mini-games* as ideias propostas por Nielsen (2011) foram consideradas. Os botões são grandes e a interface é clara. Mesmo tornando o desafio mais fácil, é importante que se mantenha os padrões de interface facilmente utilizáveis pelos usuários. Também é importante ressaltar que os eventos de *mini-game* causam uma quebra de ritmo de jogo, portanto, torná-los muito difíceis pode ser frustrante para quem joga.

Todos os eventos possuem alguma possibilidade de configuração, seja o próprio texto a ser editado, ou, no caso do evento de toque, o número de toques que o jogador deve acertar para “vencer” a etapa. Além disso, todos os eventos apresentam a opção “sensibilidade”, que representa a emoção que o evento deve causar ao jogador.

Este sistema de caminhos é uma aplicação da ideia de se escrever uma história com múltiplos caminhos e múltiplos finais. Narrativas lineares podem ser interessantes, mas limitariam muito o contexto de um jogo focado em criar histórias. Ainda assim, o jogador vai poder construir histórias com um único caminho, embora o mais natural deva construir múltiplas possibilidades.

5.2 Sensibilidade e qualidade emotiva

Quando um autor propõe uma história, ele causa emoções, sejam elas da natureza que forem. Segundo (Araújo e Ramalho, 2011) “O autor, através do mito (a história em si) que envolve personagens em conflito, comunica sua mensagem à platéia inspirando emoções como piedade e terror”. Dada a profundidade de uma história, ela pode ser capaz de comunicar muitos tipos de sentimentos e sensações aos leitores.

As emoções percebidas através de uma história vão ser diferentes para cada indivíduo que chegar a conhecê-la. Porém, partindo-se de alguns estímulos, é possível aproximar-se mais da emoção desejada utilizando-se de alguns recursos. Segundo Scherers (2011, tradução do autor) “emoções surgem quando acontece alguma coisa que o organismo considera importante, por estar diretamente ligado aos sentidos, objetivos, valores e bem estar geral da pessoa”. Portanto uma a história que consegue atingir mais sentidos de uma pessoa, pode ter mais impacto emocional. O **Story Teller**, por sua natureza digital, pode se aproveitar de recursos dinâmicos como trilha sonora e imagem. Naturalmente, por ser um software de menor porte o jogo possui produções condizentes com o seu tamanho.

O **Story Teller** possui uma trilha sonora dinâmica, que vai mudando conforme a sensibilidade escolhida para o evento. Segundo Mohn, Argstatter e Wilker (2010, tradução do autor) “a música é muito utilizada, atualmente, para causar emoções”. Os mesmos autores afirmam em seu estudo que todas as pessoas são capazes de identificar os sentimentos sendo expressados em músicas, especialmente os sentimentos de felicidade e tristeza. As variações de tempo, volume, escalas, instrumentos e distorções causam tipos de emoções diferentes.

Para o jogo desenvolvido neste trabalho construiu-se uma única música com variações sonoras menos bruscas (sem variação de tempo e instrumentos) para que o fluxo de jogo ficasse menos perturbado por variações exageradas. As variações propostas envolvem distorções e variações de ritmo.

Outro fator que ajuda a expressar a emoção do evento é a imagem do personagem principal como pode ser visto nas Figuras 16 e 20. Durante a apresentação da história o software vai apresentar ao jogador a expressão do personagem de acordo com a sensibilidade do evento. Segundo Mohn, Argstatter e Wilker (2010) as expressões faciais podem ser facilmente identificadas por qualquer pessoa.

Mais um item importante para desencadear a manifestação de emoções no jogador é o uso de cores. As cores podem possuir muitos significados e causam uma percepção emotiva que pode ser bastante forte. O seguinte trecho indica como as cores são percebidas pelas pessoas:

> "A cor exerce uma ação tríplice: a de **impressionar**, a de **expressar**, e a de **construir**. A cor é vista: impressiona a retina. E sentida: provoca uma emoção. E é construtiva, pois, tendo um significado próprio, tem valor de símbolo e capacidade, portanto, de construir uma linguagem própria que comunique uma idéia." (Farina, Perez e Bastos, 2006)

Baseado nas descrições de significados das cores de Farina, Perez e Bastos (2006) foi definido que as opções de sensibilidade do **Story Teller** terão as seguintes cores:

- **Normal (Branco)**: a cor possui um significado de neutralidade e pureza. Está associada a limpeza e clareza.
- **Calmo (Verde)**: a cor tem relação forte com natureza, mas também está associada a calma, frescor e equilíbrio.
- **Feliz/Rindo (Amarelo)**: a cor amarela está associada ao verão e à iluminação. Em comparação com tonalidades avermelhadas é uma cor menos agressiva.
- **Tenso (Vermelho escuro)**: a cor vermelha significa dinamismo e força, mas também está associada a fúria e revolta. Uma tonalidade mais escura dá uma sensação mais sombria, passando a sensação de tensão.

- **Triste/Chorando (Cinza)**: a cor dos dias chuvosos. Passa sensação de tédio, velhice, tristeza.
- **Surpreso (Laranja)**: a cor pode significar ofensa e agressão, mas também é a cor da euforia e da sexualidade. Está entre o amarelo e o vermelho e é a cor que mais possui significados misturados, portanto excelente para expressar surpresa.

Estes recursos somados devem produzir uma qualidade emocional o suficiente para que o criador das histórias possa passar a emoção que deseja ao criar a história.

5.3 Recursos adicionais do projeto

O jogo possui como plataforma de destino o iOS, sistema operacional de celulares e tablets da *Apple*. A plataforma é bastante difundida ao redor do mundo, e, portanto, atinge pessoas de diferentes nações. Uma decisão tomada durante o processo de produção foi de tornar o software multilíngue, tornando assim as possibilidades de distribuição maiores.

Foi decidido que seriam feitas versões em português por ser a língua nativa de onde foi produzido o jogo e inglês por ser uma das línguas mais falada no mundo. Para execução das traduções utilizou-se um sistema do próprio *Cocoa Framework* que permite que seja facilmente criadas traduções. O funcionamento do sistema de tradução acontece através da utilização de arquivos contendo pares de “chave” e “valor” para cada sentença a ser traduzida.

```
000.  "TITLE"                = "Title";
001.  "DESCRIPTION"          = "Description";
002.  "CHAR_NAME"            = "Character Name";
003.  "INSERT_TITLE"         = "Insert a title";
```

Código 4 - Chave e valor em inglês

```
000.  "TITLE"                = "Titulo";
001.  "DESCRIPTION"          = "Descrição";
002.  "CHAR_NAME"            = "Nome do Personagem";
003.  "INSERT_TITLE"         = "Insira um título";
```

Código 5 - Chave e valor em Português

O próprio *framework* se encarrega de colocar os arquivos no local adequado e de encontrar o caminho para as chaves de forma fácil, além de selecionar a língua certa automaticamente de acordo com a língua configurada no próprio aparelho.

Além de possuir duas línguas o **Story Teller** conta com uma característica importante em jogos digitais que é um estilo de arte. O trabalho de ilustração possui um alto valor para jogos e, em função disso, muitos criam artes características como Braid (2008) e VVVVVV (2010). Funciona como a identidade visual, um estilo artístico faz com que seja fácil reconhecer uma referência a um determinado jogo. A arte do **Story Teller** (descrita na seção 4.1.7) dá essa identidade característica ao jogo.

Por fim, é importante mencionar que o **Story Teller** possui uma sólida base de código que justamente por ser simplificada ao máximo, é fácil de entender e de estender.

6. Resultados Obtidos

Um projeto sempre deve ser testado e avaliado afim de garantir a qualidade do produto final. Executar testes é uma forma de garantir o bom funcionamento do software, bem como de melhorar a experiência do usuário. Os testes de interface e de interação possuem grande importância pois apresentam resultados da interação do usuário com o software. Segundo Preece, Rogers e Sharp (2002, tradução do autor) “avaliação é necessária para confirmar se os usuários são capazes de usar o produto e se gostam dele”.

6.1 Planejamento dos testes

Existem métodos de avaliação de interface e interação. Para o caso do **Story Teller**, o método escolhido foi testes de usabilidade, pois considera-se que ele é capaz de gerar um resultado satisfatório, em função do envolvimento de usuários. “Testes de usabilidade envolvem medir a performance de usuários típicos em tarefas comuns do sistema a ser testado” (Preece, Rogers e Sharp, 2002, tradução do autor).

O teste foi executado em duas fases. Uma primeira avaliação com um grupo de 8 pessoas a fim de identificar os problemas mais naturais com a interface do software. E uma segunda avaliação com um grupo de 7 pessoas para confirmar que os principais problemas foram corrigidos e que o software está com a interface clara e funcional.

O teste foi baseado no *framework* de avaliação de interfaces DECIDE. Segundo Preece, Rogers e Sharp (2002) o framework é “um guia para avaliação de interface”. As mesmas autoras apresentam o guia, a ser executado na avaliação, a seguir.

1. *Determine*: Determinar os objetivos da avaliação. Porque avaliar?;
2. *Explore*: Explorar questões a serem respondidas. Questões e respostas de qualidade;
3. *Choose*: Escolher os paradigmas e as técnicas para avaliação.

4. *Identify*: Identificar fatores práticos. Selecionar os testadores, seu nível de experiência e conhecimento.
5. *Decide*: Decidir como lidar com questões éticas. Informar sobre o teste e seus objetivos e nunca expor o testados a riscos ou humilhação.
6. Evaluate: Avaliar, interpretar e apresentar os dados. Interpretar os resultados.

No caso do **StoryTeller**, a aplicação do *framework* DECIDE ocorreu da seguinte forma.

Determine: o objetivo da avaliação é verificar a facilidade da utilização da aplicação pelos usuários, especialmente por ser em uma plataforma portátil. Isso somente melhora a qualidade do software, gerando, possivelmente, melhores resultados no produto final.

Explore: as questões elaboradas foram feitas com o objetivo de obter um retorno dos testadores sobre as partes que mais poderiam causar problemas de compreensão aos potenciais usuários do software. As questões elaboradas, e detalhadas na seção 6.3, são:

- O software em geral está fácil de utilizar?
- A criação e edição de eventos está fácil de utilizar?
- Os ícones e opções de menu são fáceis de entender?
- O que você achou da qualidade das ilustrações?
- O que chamou mais atenção no jogo?
- Teve alguma dificuldade de compreensão e utilização do jogo? Citar quais:
- Cite pontos positivos e negativos da aplicação.

Choose: foi decidido utilizar o paradigma de teste de usabilidade e a técnica de testes com usuários, que consiste em colocar o usuário em contato com o software, elaborar tarefas para que este realize, observar sua utilização e obter o retorno deste usuário.

Identify: os testadores selecionados eram de diferentes áreas de conhecimento, tentando, desta forma, obter a maior diversidade de pessoas para executar os testes. Essa diversidade permite que se consiga resultados melhores do

que testando com pessoas com conhecimentos muito similares. Foram selecionados alunos com formações diferentes, com e sem experiência no uso de dispositivos móveis.

Decide: a condução dos testes foi feita de forma individual. O observador apresentou o dispositivo móvel ao avaliador, informou sobre as tarefas a serem realizadas e observou a utilização. No final da seção de avaliação, o observador solicitou ao testador o preenchimento de um formulário com questões abertas a fim de obter feedback sobre a utilização do software. Todos os usuários concordaram em participar do teste e o software em si não contém nenhum fator que possa gerar problemas éticos.

Evaluate: os resultados levaram a dados qualitativos, que foram analisados e utilizados para um refinamento da interface final do software.

Na aplicação do teste cada testador executou um número determinado de tarefas que o permitiram utilizar o software de forma geral. Após concluídas as tarefas, o testador respondeu um breve questionário com o objetivo de se obter um retorno da experiência de uso. O teste foi formulado de forma que não fosse muito extenso para que os usuários se mantivessem atentos ao teste. Após análise dos questionários foi feita uma análise dos resultados obtidos.

6.2 Condução dos testes

Os testes foram executados por um grupo de usuários de diversas áreas de conhecimento, experientes com o uso de computadores e acostumados com a utilização de smartphones. Este tipo de usuário caracteriza o público alvo do **StoryTeller.** Foi decidido, também, testar com alguns usuários iniciantes no uso de smartphones pois estes são capazes de destacar uma maior quantidade de inconformidades na interface.

Devido a dificuldade da distribuição do software para vários aparelhos o teste foi aplicado individualmente com cada usuário. Apesar de tornar o processo bastante lento os testes aplicados desta forma dão a possibilidade de observação e análise mais detalhada de cada indivíduo.

A aplicação com cada usuário iniciou-se com uma descrição do que é o **StoryTeller** e quais os seus objetivos. Esta explicação deu ao usuário uma melhor compreensão das atividades que estava para executar. Após esta descrição foi dada uma lista de tarefas ao jogador (seção 6.3). Estas foram criadas para abranger a maior quantidade possível de funções do software, a ponto de que, quando o usuário terminasse de executa-las ele já conheceria o funcionamento geral do software. Desta forma, os testes mostraram resultados consistentes sobre onde estão os pontos fortes e fracos da interface do jogo.

Enquanto cada jogador procedeu com os testes, um observador estava analisando as ações tomadas por ele e verificando possíveis erros e dificuldades, bem como o tempo decorrido. Quando o jogador concluiu todas as tarefas propostas ele respondeu um breve questionário (indicado na seção 6.3) para posterior análise. Além das perguntas presentes no questionário, os usuários informaram alguns dados pessoais e informações sobre experiência com softwares em *smartphones*.

6.3 Instrumento de avaliação elaborado

O instrumento de avaliação contém as seguintes tarefas:

- Criar uma história protagonizada por uma garota chamada “Ana” e cujo título é “Um dia com Ana” e a descrição é “Uma história divertida sobre a vida da Ana”.
- Criar eventos no decorrer da história de forma que a mesma possa ter múltiplos finais.
- Visualizar a história até o encerramento pelo menos uma vez.
- Abrir uma história já criada (salva) e visualizá-la/executá-la

Estas tarefas abrangem a maior parte do software, deixando apenas alguns detalhes que não atrapalham na utilização de nenhum recurso. Os resultados foram claros quanto a problemas na interface do jogo.

Após executadas as tarefas, o testador respondeu as seguintes questões:

- O software em geral está fácil de utilizar?
- A criação e edição de eventos está fácil de utilizar?

- Os ícones e opções de menu são fáceis de entender?
- O que você achou da qualidade das ilustrações?
- O que chamou mais atenção no jogo?
- Teve alguma dificuldade de compreensão e utilização do jogo? Citar quais:
- Cite pontos positivos da aplicação:
- Cite pontos negativos da aplicação:

Estas questões foram definidas pois provêm retorno suficiente para entender a experiência que o usuário teve utilizando o software.

O Anexo B apresenta transcrições das respostas dadas pelos usuários às questões do teste de interface.

6.4 Resultados obtidos na primeira fase de testes

Os usuários que fizeram a avaliação de interface foram todos voluntários, com média de idade em torno de 22 anos. Todos eram experientes na utilização de computadores, porém apenas metade já possuía experiência com *smartphones*. Quatro deles possuíam formação voltada a ciências exatas, e outros quatro estudantes das áreas humanas.

Após feitos os testes constataram-se alguns fatos que ajudam a entender a diferença de resultados entre os diferentes usuários. O primeiro fator que identifica a facilidade de uso é a experiência do usuário com o aparelho. O tempo de execução das tarefas se torna menor conforme a experiência, visto que o que mais demorou a ser feito foi digitar os textos no teclado virtual. Outro fator que foi percebido é que os usuários que utilizam o indicador para interagir com os botões possuem mais destreza do que os que dão preferência ao polegar. Isso fica claro especialmente onde os botões são menores. Um fator aparentemente irrelevante tornou os testes um pouco mais complicados, visto que alguns usuários pareciam sentir medo de estragar e receavam tocar os botões da interface, muitas vezes parando por vários segundos até decidir qual ação tomar. Foi percebido também que todos os usuários demoraram um pouco para entender, mas enquanto usavam iam aprendendo de forma natural. Mesmo assim, algumas melhorias de interface são necessárias.

Abaixo estão listados os dados brutos da pesquisa:

- 37% dos usuários tiveram problemas com o fluxograma. O problema parece ser a falta de informações que indique como vão se abrir dois caminhos para um evento.
- 25% dos usuários tiveram problemas com tipos de eventos. Aparentemente a falta de uma descrição sobre o evento torna difícil a sua compreensão. Os usuários não tentam utilizar o recurso para ver o resultado, apenas não utilizam quando não entendem.
- 37% dos usuários tiveram problemas com o teclado virtual. Os usuários que não tinham muita experiência com smartphones foram os que mais sentiram dificuldades na utilização do teclado virtual.
- 37% dos usuários tiveram problemas com algum ícone. O ícone que mais gerou confusão, aparentemente, foi o de ícone de informações, que foi interpretado como referente à ajuda.
- 100% dos usuários elogiaram as ilustrações. O trabalho de ilustração obteve um ótimo resultado.
- 37% dos usuários destacaram a trilha sonora, apesar de não haver nenhuma questão específica sobre a arte musical. O trabalho de composição musical obteve um bom resultado.
- Os usuários levaram em média 7 minutos para executar as tarefas.

Com os resultados da avaliação foram identificados três principais correções a serem feitas:

- O ícone de informações da história precisa ser substituído utilizando uma metáfora que indique melhor sua função de informação do tipo de história.
- Uma descrição para os tipos de eventos durante a criação dos mesmos é necessária para entender o seu funcionamento.
- Uma mudança na forma de abrir dois caminhos no fluxograma. Deve ficar mais claro para o usuário quais os eventos que abrem ou não dois caminhos.

6.5 Resultados obtidos na segunda fase de testes

Os testadores desta segunda etapa também foram voluntários, porém, em oposição a primeira etapa, eram menos experientes no uso de *smartphones*, apesar de serem experientes no uso de computadores pessoais. A faixa etária ficou um pouco mais alta, em torno de 26 anos. Sobre as formações, desta vez foram entrevistados dois usuários com formação em ciências exatas, quatro com formações nas áreas humanas e um estudante de saúde.

Depois de executadas as alterações de interface o software foi testado novamente, desta vez com 7 indivíduos. Alguns dos usuários eram completamente iniciantes, porém todos concluíram as tarefas propostas sem grandes problemas.

- 100% dos usuários elogiaram as ilustrações.
- 57% dos usuários destacaram o quanto a ideia parecia interessante.
- 42% dos usuários indicaram que gostariam de um manual.
- 28% dos usuários continuaram sentindo dificuldade na compreensão do fluxograma.

Novamente, todos elogiaram as ilustrações, porém alguns ainda sentiram dificuldades com a apresentação do fluxograma. Analisando os resultados foi observado que os mais iniciantes apresentaram as maiores dificuldades, enquanto os mais experientes executaram as tarefas quase que naturalmente, desta vez. Um dos motivos que pode ter levado os usuários a sentirem dificuldades de entender o fluxograma é o seu tipo de formação. É possível que, por eles não estarem acostumados ao raciocínio estruturado, tão comum na área da computação, eles demorem mais para entender. O tempo médio de execução ficou parecido com o anterior e, novamente, o teclado virtual é o responsável.

Analisando-se especificamente os problemas apresentados anteriormente foi observado que não houve dificuldade na compreensão dos ícones. Os eventos também tiveram melhor compreensão por parte dos testadores. A descrição removeu a necessidade de testar para visualizar o evento, e deu lugar à curiosidade de ver como é o funcionamento de cada evento. O fluxograma, por outro lado, continua complexo para algumas pessoas, embora a nova forma de adicionar

eventos tenha tornado mais facilmente compreensível, alguns usuários ainda tiveram dificuldades para compreender o funcionamento do fluxograma.

É importante destacar, também, que alguns usuários solicitaram um manual que acompanhasse o software, mesmo que na forma digital. Embora seja uma possibilidade válida considera-se um manual algo excessivamente complexo para o tamanho da aplicação em questão.

Por fim, alguns testadores elogiaram a ideia do **StoryTeller**. Isso ajuda a provar que as pessoas realmente tem interesse em escrever histórias e criar conteúdo.

As figuras 21 e 22, abaixo, representam as alterações de interface feitas após a primeira fase e que foi utilizada em toda a segunda etapa de testes. Estas imagens mostram as telas que sofreram alterações do projeto original e representam o estado final do software apresentado neste projeto.

Figura 21 - Novo ícone de informações, e novos ícones de adição de evento no fluxograma

Figura 22 - Texto de descrição de evento

7. Considerações finais

O desenvolvimento de um software raramente segue um processo sem iteração. A tarefa é repleta de problemas e soluções escondidas que tornam a construção de um produto uma tarefa de alto nível de complexidade. O **Story Teller** não é exceção e sofreu mudanças necessárias durante o desenvolvimento. Normalmente as maiores mudanças se dão por falta de uma melhor análise, ou simplesmente por falta de experiência por parte da equipe de desenvolvimento. Considerando tudo, este projeto possui características experimentais, e portanto não é surpreendente ver que a falta de experiência com este tipo de desenvolvimento leva a alterações no procedimento de construção do software.

Todo o processo de desenvolvimento desde a seleção de tecnologia até a produção do software foi repleto de descobertas. A falta de experiência especialmente com a plataforma gerou muitos problemas que foram sendo contornados aos poucos. Felizmente a escolha do Cocos2D facilitou bastante a produção do **Story Teller**. Infelizmente trabalhar com dispositivos móveis foi uma experiência bem abaixo do esperado. Houve excessivas complicações de coisas que pareciam ser simples, como a instalação do software num aparelho de teste. As imposições que foram feitas em termos de recursos técnicos para desenvolvimento também foram desagradáveis, pois a falta de documentação de desenvolvimento impediu que o trabalho pudesse ser produzido fora da IDE (Ambiente Integrado de Desenvolvimento) padrão para o iOS. Tudo isso fez com que, em muitas situações, fosse necessário despender muito tempo para resolver problemas estruturais em vez de desenvolver o software.

Porém, acima de qualquer possível falha de análise ou projeto os resultados obtidos foram totalmente satisfatórios, mostrando que mesmo com alguns desvios é possível construir um software funcional, desde que se esteja preparado para tal.

As descobertas e o aprendizado neste trabalho foram fantásticos. O conhecimento adquirido foi além das expectativas e provou que com esforço é possível chegar até onde não se espera. Além disso, a possibilidade de desenvolver um software seguindo um rigor metodológico, desde a documentação do processo até o planejamento e a execução dos testes, ampliou vastamente a visão do

processo de desenvolvimento de um software. Foi possível entender a importância do processo para agregar qualidade em um produto de software.

O futuro do **Story Teller** está em aberto. Há algumas possibilidades claras para se implementar:

- Melhora no sistema de transição das músicas: quando se passa de um evento para outro é possível fazer com que as duas partes da trilha se mesclem de uma forma que não exista (ou que fique imperceptível) a quebra de ritmo.
- Adição de novos tipos de eventos e especialmente minigames: aumentar a quantidade de possibilidades deve dar ao jogador muito mais liberdade criativa, podendo tornar as histórias muito mais interessantes.
- Adição de possibilidade de compartilhamento de histórias: este seria um ponto muito interessante visto que o jogador, atualmente, após construir a história, deve entregar o aparelho para a pessoa que irá conhecer o roteiro. A possibilidade de distribuir de outra forma pode tornar o **Story Teller** muito mais interessante.
- Criação de versão para outro sistema operacional: as possibilidades de utilização do software podem ser expandidas para alternativas mais baratas e mais abertas (smartphones que utilizam Android e Symbian costumam ter custo menor).

Acima de tudo, o mais importante é ser criativo e ser capaz de criar coisas novas por mais simples que possam ser.

6. Referências Bibliográficas

APPLE. **Coding Guidelines for Cocoa:** Code Naming Basics. Disponível em: <https://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/CodingGuidelines/Articles/NamingBasics.html#//apple_ref/doc/uid/20001281-BBCHBFAH>. Acesso em: 9 out. 2011.

APPLE. **Mobile Human Interface Guidelines**. Disponível em: <https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/MobileHIG.pdf>. Acesso em: 27 set. 2011.

ARAÚJO, Raony M.; RAMALHO Geber L.. **Narrativa e Jogos Digitais**: Lições do RPG de Mesa. In: SIMPÓSIO BRASILEIRO DE GAMES, 2006, Recife. Disponível em: <http://www.sbgames.org/papers/sbgames06/27.pdf>. Acesso em: 12 set. 2011.

BAL, Mieke. **Narratology**: Introduction to the theory of narrative. 2. ed. Toronto: University Of Toronto Press Incorporated, 1999.

BEATRIZ, Isa; MARTINS, Jodeilson; ALVES, Lynn. **A Crescente Presença da Narrativa nos Jogos Eletrônicos**. In: SIMPÓSIO BRASILEIRO DE GAMES E ENTRETENIMENTO DIGITAL, 8, 8 out. 2009, Rio de Janeiro. Disponível em: <http://www.sbgames.org/papers/sbgames09/culture/full/cult2_09.pdf>. Acesso em: 8 ago. 2011.

BENYON, David. **Interação Humano-Computador**. 2. ed. São Paulo: Pearson Education do Brasil, 2011.

CATTO, Erin. **Box2D User Manual**. Disponível em: <http://box2d.org/manual.pdf>. Acesso em: 5 out. 2011.

COCOS2D. **Cocos2D iPhone**. Disponível em: <http://www.cocos2d-iphone.org/about>. Acesso em: 6 out. 2011.

CONSALVO, Mia. Console video games and global corporations: Creating a hybrid culture. **New Media & Society**, Londres, p.117-137, 2006. Disponível em: <http://nms.sagepub.com/content/8/1/117.abstract>. Acesso em: 21 set. 2011.

CRAWFORD, Chris. **Chris Crawford on Interactive Storytelling**. Berkeley: New Riders, 2005.

CROOK, Jordan. **Mobile gaming market to reach $18B by 2014:** Study. Publicado em 11 ago 2009. Disponível em: <http://www.mobilemarketer.com/cms/news/research/3892.html>. Acesso em: 3 out. 2011.

FARINA, Modesto; PEREZ, Clotilde; BASTOS, Dorinho. **Psicodinâmica das Cores em Comunicação**. 5. ed. São Paulo: Edgarg Blücher Ltda., 2006.

FLYNT, John; SALEM, Omar. **Software Engineering for Game Developers**. Boston: Thomson, 2005.

FMOD. **FMod:** interactive audio middleware. Disponível em: <http://www.fmod.org/index.php/products/fmodex>. Acesso em: 5 out. 2011.

GREGORY, Jason. **Game Engine Architecture**. Estados Unidos da América: Taylor And Francis Group, 2009.

GUINESS WORLD RECORDS. **Guiness World Records 2009 Games**. São Paulo: Ediouro, 2009.

HERMAN, Luc; VERVAECK, Bart. **Handbook of Narrative Analysis:** Frontiers of Narrative. United States Of America: University Of Nebraska Press, 2005.

INFAMOUSTHEGAME. **Infamous the Game**. Disponível em: <http://infamousthegame.software.eu.playstation.com/old/#/en_US/features>. Acesso em: 25 nov. 2011.

IP, Barry. Narrative Structures in Computer and Video Games: Part 1: Context, Definitions, and Initial Findings. **Games And Culture**, [s. l.], p.103-134, 7 maio 2010. Disponível em: <http://gac.sagepub.com/content/6/2/103>. Acesso em: 15 ago. 2011.

KILLEN, Tom. **The Mobile Market Place:** How Does It Look Today?. Publicado em 26 set 2011. Disponível em: <http://www.gamasutra.com/blogs/TomKillen/20110926/8520/The_Mobile_Market_Place_How_Does_It_Look_Today.php>. Acesso em: 3 out. 2011.

KOOLSTRA, Cees ; BOS, Mark. The Development of an Instrument To Determine Different Levels of Interactivity. **International Communication Gazette**, [s. L.], p. 373-391, 2009. Disponível em: <http://gaz.sagepub.com/content/71/5/373>. Acesso em: 16 ago. 2011.

LEBOWITZ, Josiah; KLUG, Chris. **Interactive Storytelling for Video Games:** A Player-Centered Approach to Creating Memorable Characters and Stories. Burlington: Elsevier, 2011.

LESNOVSKI, Ana Flávia Merino. **Por Baixo dos Telhados:** O Modelo da Simulação como Mecanismo para a Narrativa Interativa. In: GAMEPAD: SEMINÁRIOS DE GAMES, COMUNICAÇÃO E TECNOLOGIA. 28, maio 2011, Novo Hamburgo.

MCSHAFFRY, Mike; et al. **Game Coding Complete:** Third Edition. 3 ed. Boston: Course Technology, a part of Cengage Learning, 2009.

MCREYNOLDS, Tom; BLYTHE David. **Advanced Graphics Programming Using OpenGL**. São Francisco: Elsevier, 2005.

MOHN, Christine; ARGSTATTER, Heike; WILKER, Friedrich-wilhelm. Perception of six basic emotions in music. **Psychology Of Music**, S. L., n. , p.503-517, out. 2010. Disponível em: <http://pom.sagepub.com/content/39/4/503>. Acesso em: 14 mar. 2012.

NIELSEN, Jakob. **Mobile Usability Update**. Publicado em 26 set 2011. Disponível em: <http://www.useit.com/alertbox/mobile-usability.html>. Acesso em: 4 out. 2011.

NOVAK, Jeannie. **Desenvolvimento de Games**: Tradução da 2ª Edição Norte-Americana. São Paulo: Cengage Learning, 2010. 443 p.

PERUCIA, Alexandre Sousa et al. **Desenvolvimento de Jogos Eletrônicos**: Teoria e Prática. 2. ed. São Paulo: Novatec, 2005.

PREECE, Jennifer; ROGERS, Yvonne; SHARP, Helen. **Interaction Design**: Beyond Human-Computer Interaction. S. L: John Wiley & Sons, Inc., 2002.

RAENTO, Mika. Smartphones: An Emerging Tool for Social Scientists. **Sociological Methods & Research**, Londres, n. , p.426-454, fev. 2009. Disponível em: <http://smr.sagepub.com/content/37/3/426>. Acesso em: 4 out. 2011.

ROLLINGS, Andrew; MORRIS, Dave. **Game Architecture and Design:** A New Edition. 2. ed. Indianapolis: New Riders, 2004.

SCHERERS, Klaus R. On the rationality of emotions: or, When are emotions rational?. **Social Science Information**, S. L., n. , p.330-350, ago. 2011. Disponível em: <http://ssi.sagepub.com/content/50/3-4/330>. Acesso em: 13 mar. 2012.

SDL. **Simple Directmedia Layer**. Disponível em: <http://www.libsdl.org/>. Acesso em: 6 out. 2011

SHOKOUHI, Hossein; DARAM, Mahmood; SABAH, Somayeh. Shifting between third and first person points of view in EFL narratives. **Arts And Humanities In Higher Education**, [s. L.], n. , 20 jun. 2011. Disponível em: <http://ahh.sagepub.com/content/early/2011/06/15/1474022210386573>. Acesso em: 20 set. 2011.

SQUARE. **Final Fantasy Tactics**: The War of the Lions. Disponível em: <http://na.square-enix.com/fftactics/>. Acesso em: 22 set. 2011.

STROUGO, Rod; WENDERLICH, Ray. **Learning Cocos 2D:** A Hands-On Guide to Building iOS Games with Cocos2D, Box2D and Chipmunk. Boston: Addison-wesley, 2011.

THORN, Alan. **Game Engine Design and Implementation**. Canada: Jones & Barlett Learning, 2011.

UNITY3D. **Unity3D Game Development Tool**. Disponível em: <http://unity3d.com/unity/>. Acesso em: 6 out. 2011.

WRIGHT JUNIOR, Richard; LIPCHAK, Benjamin; HAEMEL, Nicholas. **OpenGL SuperBible:** Comprehensive Tutorial and Reference. 4. ed. Boston: Pearson Education Inc., 2007.

ZERBST, Stefan; DÜVEL, Oliver. **3D Game Engine Programming**. United States Of America: Premier Press, 2004.

Referências de Jogos

Braid. Desenvolvedor Number None Inc. e Hothead Games. Publicador Microsoft Games Studios e Number None Inc. 2008.

Chrono Cross. Desenvolvedor Square. Publicador Square. 2000.

Colossal Cave Adventure. Desenvolvedor William Crowther e Don Woods. Publicador CRL. 1976.

Disgaea. Desenvolvedor Nippon Ichi Software. Publicadores Nippon Ichi Software, Atlus, Square Enix e Ubisoft. 2003.

Final Fantasy Tactics. Desenvolvedor Square. Publicador Sony Computer Entertainment. 1998.

Final Fantasy XIII. Desenvolvedor Square Enix. Publicador Square Enix. 2010.

Frogger. Desenvolvedor Konami. Publicador Sega. 1981.

inFAMOUS. Desenvolvedor Sucker Punch Productions. Publicador Sony Computer Entertainment. 2009.

Little Big Planet. Desenvolvedor Media Molecule. Publicador Sony Computer Entertainment. 2008.

Minercraft. Desenvolvedor Mojang. Publicador Mojang. 2009.

ModNation Racers. Desenvolvedor United Front Games. Publicador Sony Computer Entertainment. 2010.

Pacman. Desenvolvedor Namco. Publicador Namco. 1980.

Phoenix Wright. Desenvolvedor Capcom. Publicador Capcom. 2001.

Tetris. Desenvolvedor Alexey Pajitnov. Publicador varios. 1984.

The Sims 3. Desenvolvedor The Sims Studio. Publicador Eletronic Arts. 2009.

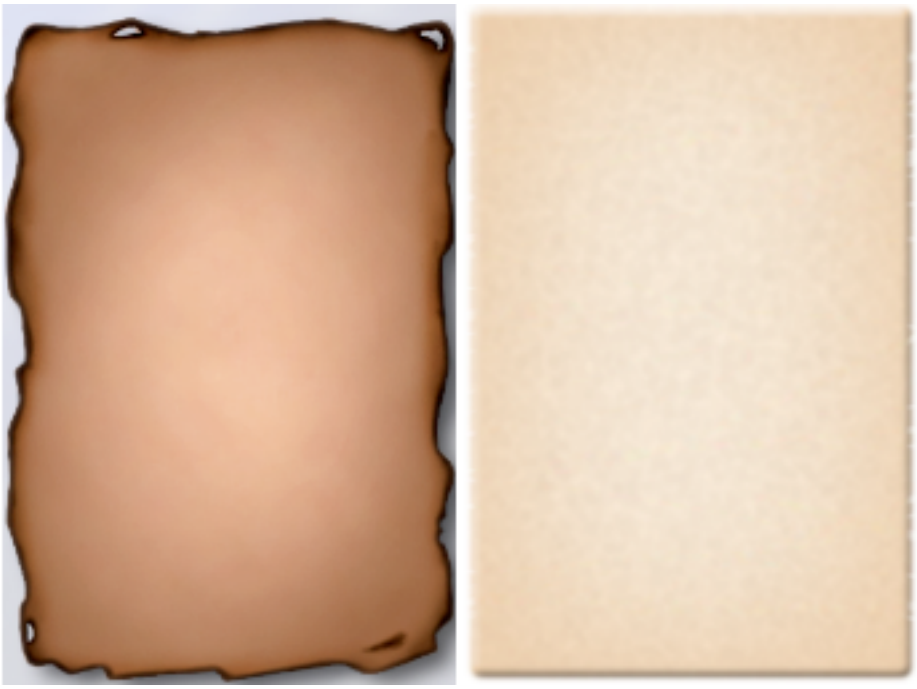
White Knight Chronicles. Desenvolvedor Japan Studio. Publicador Sony Computer Entertainment. 2010.

VVVVVV. Desenvolvedor Terry Cavanagh. Publicador Nicalis. 2010

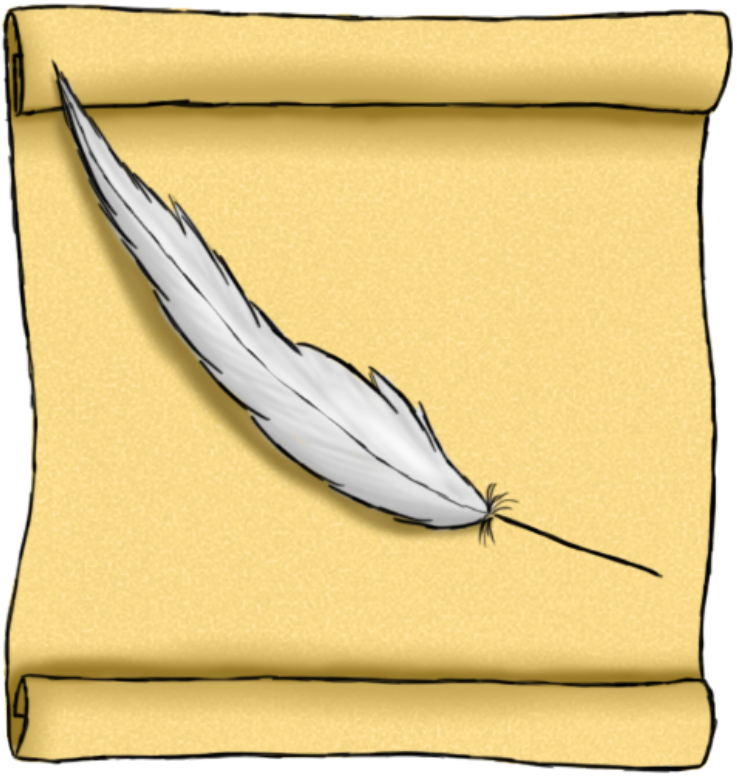
Anexo A

As imagens abaixo foram produzidas por Anne Lorandi Pagno especialmente para este trabalho.

Personagens e suas expressões



Fundos de tela



Ícone do jogo

Anexo B

Transcrição dos questionários aplicados aos testadores do software.

Primeiro teste:

Testador 1

O software em geral está fácil de utilizar?

R: Sim, porém o fluxograma parece muito "automático". Demorei para entender.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim, bastante fáceis.

O que você achou da qualidade das ilustrações?

R: Tudo muito bonito!

O que chamou mais atenção no Jogo?

R: Os personagens.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Achei o fluxograma "automático demais".

Cite os pontos positivos da aplicação.

R: Gostei dos personagens, do fato de poder criar uma história, de poder escolher as sensações e da música.

Cite os pontos negativos da aplicação.

R: Fluxograma automático.

Testador 2

O software em geral está fácil de utilizar?

R: Sim, mas apesar de simples demorei para entender como funciona.

A criação e edição de eventos está fácil de utilizar?

R: Não muito. Não se sabe o que cada tipo de evento faz.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Bonitas!

O que chamou mais atenção no Jogo?

R: A arte e a música.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: O que cada tipo de evento faz.

Cite os pontos positivos da aplicação.

R: A ideia de se criar histórias.

Cite os pontos negativos da aplicação.

R: Touch não está fácil de usar.

Testador 3

O software em geral está fácil de utilizar?

R: Sim.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Bacana!

O que chamou mais atenção no Jogo?

R: O evento de final do jogo, e a forma de apresentação da história ficou além do que esperava.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Não

Cite os pontos positivos da aplicação.

R: A música.

Cite os pontos negativos da aplicação.

R: Teclado virtual.

Testador 4

O software em geral está fácil de utilizar?

R: Não. Não consegui entender o fluxograma de primeira.

A criação e edição de eventos está fácil de utilizar?

R: A princípio não, mas consegui entender usando.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Legal.

O que chamou mais atenção no Jogo?

R: Ilusrações e "as coisas se mexendo na apresentação".

Teve alguma dificuldade de compreensão e utilização do jogo?

R: O fluxograma.

Cite os pontos positivos da aplicação.

R: Os desenhos.

Cite os pontos negativos da aplicação.

R: Evento de toque! (Não gostou porque teve que usar pra entender o que era)

Testador 5

O software em geral está fácil de utilizar?

R: Sim

A criação e edição de eventos está fácil de utilizar?

R: Sim

Os icones e opções de menu são fáceis de entender?

R: Mais ou menos. O i (ícone de informações) dá a impressão de ser um “ajuda”.

O que você achou da qualidade das ilustrações?

R: Bem legal!

O que chamou mais atenção no Jogo?

R: Nada em especial.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Demorei a entender o i (ícone de informações)

Cite os pontos positivos da aplicação.

R: Ilustrações e interface disposição da interface em geral.

Cite os pontos negativos da aplicação.

R: i (ícone de informações) e o teclado virtual

Testador 6

O software em geral está fácil de utilizar?

R: “Médio”. Fica fácil pra quem está acostumado com coisas de raciocínio lógico.

A criação e edição de eventos está fácil de utilizar?

R: Sim

Os icones e opções de menu são fáceis de entender?

R: O ícone de adição de evento (+) não deu de entender a princípio.

O que você achou da qualidade das ilustrações?

R: Ficou bom, cara!

O que chamou mais atenção no Jogo?

R: Possibilidade de se divertir por algumas horas.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: O ícone de adição de evento (+)

Cite os pontos positivos da aplicação.

R: Idéia simples e fácil.

Cite os pontos negativos da aplicação.

R: O ícone de adição de evento (+)

Testador 7

O software em geral está fácil de utilizar?

R: Há uma dificuldade natural por ser a primeira vez que se utiliza.

A criação e edição de eventos está fácil de utilizar?

R: Confuso na hora de abrir 2 caminhos.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Bom. Ficou muito bom!

O que chamou mais atenção no Jogo?

R: Nada em especial

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Como abrir 2 caminhos.

Cite os pontos positivos da aplicação.

R: Pareceu muito bem feito. Gostei da organização e tamanho dos ícones.

Cite os pontos negativos da aplicação.

R: Nada

Testador 8

O software em geral está fácil de utilizar?

R: Sim.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Bonitas. Gostei dos personagens.

O que chamou mais atenção no Jogo?

R: Os desenhos.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Me atrapalhei no início, até entender como fazer dois finais.

Cite os pontos positivos da aplicação.

R: Desenhos!

Cite os pontos negativos da aplicação.

R: Teclado Virtual.

Segundo teste:

Testador 9

O software em geral está fácil de utilizar?

R: Não o suficiente. Me perdi as vezes.

A criação e edição de eventos está fácil de utilizar?

R: Não entendi bem como funciona até começar a usar.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Legal.

O que chamou mais atenção no Jogo?

R: A idéia de construir histórias e a sensibilidade dos eventos.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Senti falta de uma explicação.

Cite os pontos positivos da aplicação.

R: Nada em especial.

Cite os pontos negativos da aplicação.

R: Nada em especial.

Testador 10

O software em geral está fácil de utilizar?

R: Um manual ajudaria.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Muito boas!

O que chamou mais atenção no Jogo?

R: A idéia do construir histórias.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Faltou um manual. Não achei os botões muito intuitivos.

Cite os pontos positivos da aplicação.

R: Os gráficos (imagens).

Cite os pontos negativos da aplicação.

R: Teclado Virtual.

Testador 11

O software em geral está fácil de utilizar?

R: É bastante intuitivo, porém, talvez até demais.

A criação e edição de eventos está fácil de utilizar?

R: Mesma resposta da questão anterior.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Boa! Bem agradável.

O que chamou mais atenção no Jogo?

R: Idéia de criar histórias.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: O fluxograma.

Cite os pontos positivos da aplicação.

R: A idéia do app, especialmente por ser mobile.

Cite os pontos negativos da aplicação.

R: Nada em especial.

Testador 12

O software em geral está fácil de utilizar?

R: No começo é um pouco estranho

A criação e edição de eventos está fácil de utilizar?

R: Não.

Os icones e opções de menu são fáceis de entender?

R: Ok.

O que você achou da qualidade das ilustrações?

R: Bem interessante.

O que chamou mais atenção no Jogo?

R: A criatividade e inovação que representa.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Em especial, o fluxograma.

Cite os pontos positivos da aplicação.

R: A idéia do jogo, a criatividade.

Cite os pontos negativos da aplicação.

R: Faltou um tutorial ou um modelo para servir de exemplo.

Testador 13

O software em geral está fácil de utilizar?

R: Sim, não achei difícil.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Ótimas! Queria desenhar assim.

O que chamou mais atenção no Jogo?

R: Os desenhos, mesmo.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Não.

Cite os pontos positivos da aplicação.

R: Os desenhos e os mini-games.

Cite os pontos negativos da aplicação.

R: Só funciona no iPhone.

Testador 14

O software em geral está fácil de utilizar?

R: Podia ser mais fácil.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Legais.

O que chamou mais atenção no Jogo?

R: Nada em especial.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Achei estranho como adicionar eventos, mas depois me acostumei.

Cite os pontos positivos da aplicação.

R: É bonito e rápido de usar.

Cite os pontos negativos da aplicação.

R: Nada em especial.

Testador 15

O software em geral está fácil de utilizar?

R: Sim.

A criação e edição de eventos está fácil de utilizar?

R: Sim.

Os icones e opções de menu são fáceis de entender?

R: Sim.

O que você achou da qualidade das ilustrações?

R: Boas.

O que chamou mais atenção no Jogo?

R: Tem música.

Teve alguma dificuldade de compreensão e utilização do jogo?

R: Edição de título, descrição e nome do personagem. Não parecem botão de edição.

Cite os pontos positivos da aplicação.

R: Pode ser bem divertido.

Cite os pontos negativos da aplicação.

R: Não dá pra usar com o iPhone “de lado”.