UNIVERSIDADE FEDERAL DE PERNAMBUCO
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
CENTRO DE INFORMÁTICA

UM ESTUDO COMPARATIVO DE TECNOLOGIAS WEB PARA GAMES

TRABALHO DE GRADUAÇÃO

Aluno: Mário Barbosa de Araújo Júnior (mbaj@cin.ufpe.br)
Professor: Geber Lisboa Ramalho (glr@cin.ufpe.br)

Recife, PE
2009

Assinaturas

Este Trabalho de Graduação é resultado dos esforços do aluno Mário Barbosa, sob a orientação do professor Geber Ramalho, sob um titulo de “Um Estudo Comparativo de Tecnologias Web Para Games”. Todos abaixo estão de acordo com o conteúdo deste documento e os resultados deste Trabalho de Graduação.

_____

Mário Barbosa de Araújo Júnior

_____

Geber Lisboa Ramalho

"O único lugar onde sucesso vem antes de trabalho é no dicionário"

(Albert Einstein)

Agradecimentos

Agradeço aos meus pais (Mário e Edjane) que sempre colocaram a educação em primeiro lugar, priorizando sempre os estudos e dando o maior apoio nos momentos de dificuldade. E principalmente a minha irmã Bruna que me atura desde que veio ao mundo.

Agradeço a minha namorada e amiga Pollyanna que me acompanhou por praticamente toda a minha vida acadêmica, que soube superar os momentos de ausência e stress, que sempre esteve ao meu lado nas decisões mais importantes da minha vida. E hoje em dia até deixa eu virar noite no CIn sem ficar muito brava.

Agradeço ao pessoal da Educandus, João Augusto, Clóvis, Renato, David, Bernardo, Bruna, Leonardo, Priscila, Valeska, Luis, Cynthia, Bruno Ramirez, Cleano e muita gente que passou por lá, principalmente a Ricardo Lessa, meu ex-chefe que acreditou no meu potencial e me deu oportunidade de trabalhar nessa grande família que é a Educandus, cada um ajudou um pouco em formar a minha personalidade e o meu ciclo de amizade.

Um agradecimento especial para os meu mentores, Tairone, Laís, Teoria, Ricardo, Felipe e Luciano. Foram eles os principais responsáveis pela minha formação como profissional.

Sou muito grato também ao pessoal da FAST. Foi um momento curto que passei por lá, mais obtive um ciclo de amizade para a vida toda.

Agradeço ao pessoas da i2 tecnologia, Thiago, Felipe, Felipe Andrade, Luciano, Edson, Wiviane, Matheus e Riroshi. Principalmente Felipe que veio segurando a barra com os clientes pela minha ausência para a escrita deste trabalho.

E finalmente um agradecimento especial ao pessoal do China 48, Xumiga, Apebão, Peter, Doido, Lhama Soso, JP e Riffa. Essa galera me acompanhou por todo o curso e continuará me acompanhando na vida profissional e na amizade.

Agradeço a todos que me acompanhou durante essa fase trabalhosa mas de grandes resultados e crescimento pessoal. E principalmente a você que esta lendo essa parte do trabalho que todo mundo pula.

Resumo

Este trabalho tem como objetivo auxiliar o desenvolvedor na escolha de uma plataforma para desenvolvimento de jogos na web, apresentando as principais plataformas para desenvolvimento bem como suas vantagens e desvantagens e suas principais características. Dentre as plataformas analisadas foram escolhidas duas para um estudo comparativo mais técnico, e concluindo qual delas é mais adequada para determinada situação durante o desenvolvimento.

Para fazer o cruzamento de dados, também foi desenvolvido um framework de comparação, onde foi definindo as variáveis e os critérios de avaliação que iriam se relacionar, e como foi aferido o desempenho e complexidade de cada uma das tecnologias. E por fim duas tecnologias foram escolhidas para o estudo de caso, sendo desenvolvido o mesmo jogo nas duas plataformas de desenvolvimento, e com base nas experiências e dificuldades encontradas no desenvolvimento de cada jogo uma tabela comparativas foi preenchida relatando essas experiências.

Índice

ASSINATURAS **2**

AGRADECIMENTOS **4**

RESUMO **6**

ÍNDICE **7**

1. INTRODUÇÃO **9**

1.1 PROBLEMAS E JUSTIFICATIVAS 9

1.2 OBJETIVO 9

1.4 ESTRUTURA GERAL 10

2 JOGOS NO BROWSER **11**

2.1 MERCADO 12

2.2 O QUE SE ESPERAR DE UMA PLATAFORMA DE DESENVOLVIMENTO? 14

3. ESTADO DA ARTE **18**

3.1 JAVA 19

3.1.1 História *19*

3.1.2 Ambiente de Execução *20*

3.1.3 Linguagem de Programação *21*

3.1.4 Desenvolvendo na Plataforma *21*

3.1.5 Java para Jogos *23*

3.2 SILVERLIGHT 24

3.2.1 História *24*

3.2.1 Ambiente de Execução *26*

3.2.2 Linguagem de Programação *26*

3.2.3 Ferramentas de Desenvolvimento *30*

3.3 FLASH 31

3.3.1 História *32*

3.3.2 Ambiente de Execução *32*

3.3.3 Linguagem de Programação *33*

3.3.4 Desenvolvimento na Plataforma *34*

3.4 ESTUDO COMPARATIVO 36

3.4.1 Facilidade *36*

3.4.2 Qualidade *41*

4. ESTUDO DE CASO **45**

4.1 ESTUDO COMPARATIVO 46
4.1.1 Fatores Externos *46*
4.1.2 Resultados do Estudo *46*
5. CONSIDERAÇÕES FINAIS 54
5.1 CONCLUSÃO 54
6.2 TRABALHOS FUTUROS 55
REFERÊNCIAS 56
APÊNDICE A – DOCUMENTO DE GAME DESIGN 59

1. Introdução

As aplicações web são aplicações de software que utilizam o browser como ambiente de execução, estas que são geralmente multiplataforma tem a vantagem de poder ser executada em qualquer máquina com acesso a internet, bastando apenas que a máquina possua os pré-requisitos desejados pela aplicação. Algumas aplicações até permitem que os arquivos dos usuários fiquem salvos na "nuvem", sem que o usuário necessite armazená-lo em nenhum meio físico.

1.1 Problemas e Justificativas

As aplicações online vêm crescendo e alimentando o mercado com novos produtos constantemente, este novo paradigma de software como serviço vem ganhando cada vez mais espaço, freqüentemente encontramos versões simplificadas online de aplicações que antes só eram encontradas no desktop, hoje existem desde aplicações de escritório como processador de textos e planilhas até editores de imagem rodando diretamente no browser, esta liberdade e facilidade de uso vem atraindo cada vez mais usuários, acabando com a necessidade de só poder trabalhar ou se divertir em uma maquina específica, podendo abrir seus documentos em qualquer lugar bastando estar conectado a internet.

Esta transição de aplicações do computador para o browser vem crescente em ritmo acelerado, cobrindo quase todas as necessidades de um usuário comum, mas nem todas as aplicações conseguem ter um substituto online a altura de uma aplicação local, as aplicações mais exigentes como os editores de vídeo e principalmente os jogos necessitam de mais recursos que muitas tecnologias web não consegue prover, necessitando de tecnologias especiais para cumprir o propósito.

1.2 Objetivo

O objetivo deste trabalho é apresentar as principais tecnologias de desenvolvimento de jogos para a web, apresentando as vantagens e desvantagens de cada tecnologia. O foco do projeto é na facilidade que cada ferramenta prove para o desenvolvedor, considerando o nível de integração com outras ferramentas, e a capacidade das plataformas de trabalhar com elementos básico de um jogo como o som, vídeo e gráficos. Sendo uma pesquisa para ajudar gerentes de projeto a escolher a tecnologia que melhor cumpre com as necessidades do projeto.

1.4 Estrutura Geral

Este trabalho esta definido em cinco capítulos e um apêndice. Neste primeiro capítulo encontra-se a descrição do trabalho, o objetivo, os problemas e justificativas para a realização do estudo. No segundo capítulo será encontrado uma descrição de jogos web, apresentando a situação atual do mercado para jogos casuais e qual a participação do Brasil neste mercado, e por fim um tópico explicando do que se esperar de uma plataforma para o desenvolvimento de jogos para a web. No capítulo 3 será apresentada, os pré-requisitos mínimos que um jogo simples na web deve possuir e quais as principais tecnologias que conseguem desenvolver este produto, com uma descrição de cada uma delas, e por fim será apresentando um estudo comparativo de acordo com as características teóricas disponibilizadas por cada fabricante de suas respectivas tecnologias. No capítulo 5 será escolhida as duas principais tecnologias para que sejam realizadas os testes prático de comparação entre elas, desenvolvendo o mesmo jogo em cada uma delas e mostrando os diferentes resultados de implementação de cada tecnologia nas mesmas situações. E por fim o capítulo de considerações finais, apresentando a melhor plataforma para cada caso abordado e sugerindo trabalhos futuros para dar continuidade a este estudo.

2 Jogos no Browser

Jogos Web são jogos que rodam diretamente no browser. Diferente dos jogos de computados os jogos web não necessitam ser instalados no computador do usuário, eles são iniciados no próprio browser. Esses jogos utilizam tecnologias client-side como JavaScript ou plug-ins como Java ou Flash, e alguns jogos utilizam também tecnologias server-side para aumentar a interatividade e o game replay, que vão desde um simples tabela de melhores *scores* do game até um game onde os jogadores interajam entre si, seja num combate ou em um modo cooperativo, aumentando assim a vida útil do game. Esses jogos que funcionam no browser são chamados de browser-based games [2].

Inicialmente os jogos browser-based eram jogos simples, geralmente jogos casuais, pois as tecnologias existentes anteriormente limitavam a complexidade dos games, mas hoje em dia, com o avanço das tecnologias baseadas em plugins para o browser existem diversas alternativas para o desenvolvimento de games complexos, mesmo assim os jogos casuais ainda dominam este mercado.

Mesmo com o avanço das tecnologias web para jogos, os jogos mais jogados ainda continuam sendo os jogos casuais. O motivo dessa dominância no mercado em parte é conseqüência das características simples dos jogos casuais, são elas [25]:

- Jogos de fácil aprendizagem
- Controles simples, com poucas teclas e de funcionamento intuitivo.
- Gameplay atraente, com incremento de conteúdo durante o jogo pequeno. Pois a maioria dos jogos casuais são jogados entre 5 a 20 minutos.
- Jogos casuais procuram apenas entreter o jogados com uma diversão relaxante, diferente dos jogos hardcore que pretende gerar adrenalina.
- Jogos casuais geralmente necessitam de baixo custo de produção em relação a jogos hardcere, reduzindo assim o tempo de retorno e o risco do investimento.

Contudo nem todos os jogos casuais possuem todas as categorias, mas todos possuem a característica mais relevante, a acessibilidade. Ou seja, jogos casuais podem ser

definidos como jogos de fácil entrada de qualquer tipo de usuário. Seja qual for a definição, uma coisa é certa, jogos casuais surgiram como uma massa de mercado de entretenimento rápido, e com novos modelos de distribuição e geração de renda diferenciada aos jogos *hardcore* [25].

2.1 Mercado

O mercado de jogos se encontra em pleno crescimento. Atualmente, é publicado um estudo chamado ESA Facts - pela Entertainment Software Association (ESA), que busca identificar os mais relevantes aspectos com relação ao desenvolvimento de software voltado para o entretenimento. Segundo a edição de 2008, os jogos casuais na internet corresponde a cerca de 61% dos jogos online jogados com mais freqüência, ficando na frente dos jogos de ação, estratégia e até dos jogos com um universo multi-player persistente como os MMORPG [1].

Segundo pesquisas do ESA Facts, 22% dos jogadores mais freqüentes pagam para jogar jogos online, esse valor cresceu 19% desde 2007. Dos tipos de jogos online jogados com mais freqüência os jogos casuais corresponde a grande maioria, cerca de 47%, estando na frente dos jogos de ação e estratégia que corresponde a 16% do total [1].

Na industria de jogos causais, no estágio mais inicial o mercado, o sucesso de um jogo é representado pela quantidade de jogadores diferentes, mas nenhum tem presença dominante no mercado [25]. Os modelos de negócios é a grande sacada na distribuição de jogos, a maioria das empresas preenchem múltiplos papeis na distribuição de suas aplicações para reduzir o custo de lançamento e manutenção de um jogo, como por exemplo, desempenhar os papeis de Desenvolver/Publicar, Distribuir/Revender ou Desenvolver/Publicar/Distribuir. O IGDA – International Game Developer Association, uma associação de desenvolvedores de jogos que serve para aproximar os profissionais da área, criando redes de discussão, e mostrando as melhores áreas do mercados a se investir, publicando artigos e pesquisas relacionadas a área, revela as estatísticas de distribuição dos royalties entre os setores do desenvolvimento de um jogo, segundo os cinco maiores portais

de distribuição de jogos casuais, a taxa de royalties esta distribuído entre 61% a 70%.[25] [26] [27]

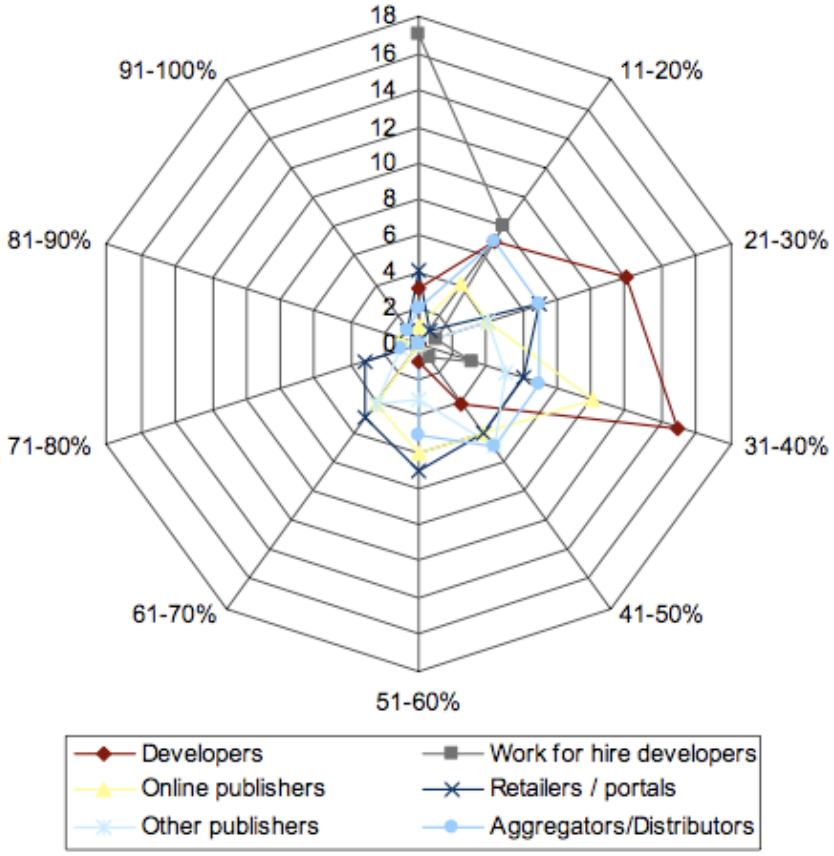


Figura 1 – Distribuição de royalty para os games casuais

Segundo um estudo publicado pela AbraGames – Associação Brasileira das Desenvolvedoras de Jogos Eletrônicos, uma instituição sem fins lucrativos que tem como principal objetivo fortalecer a industria nacional de desenvolvimento de jogos, uma pesquisa realizada em 2008 mostra que a industria nacional vem obtendo um crescimento da participação de produtos cada vez mais complexos no faturamento das empresas. [2] O gráfico a seguir mostra o crescimento do mercado brasileiro em cada segmento dos jogos.

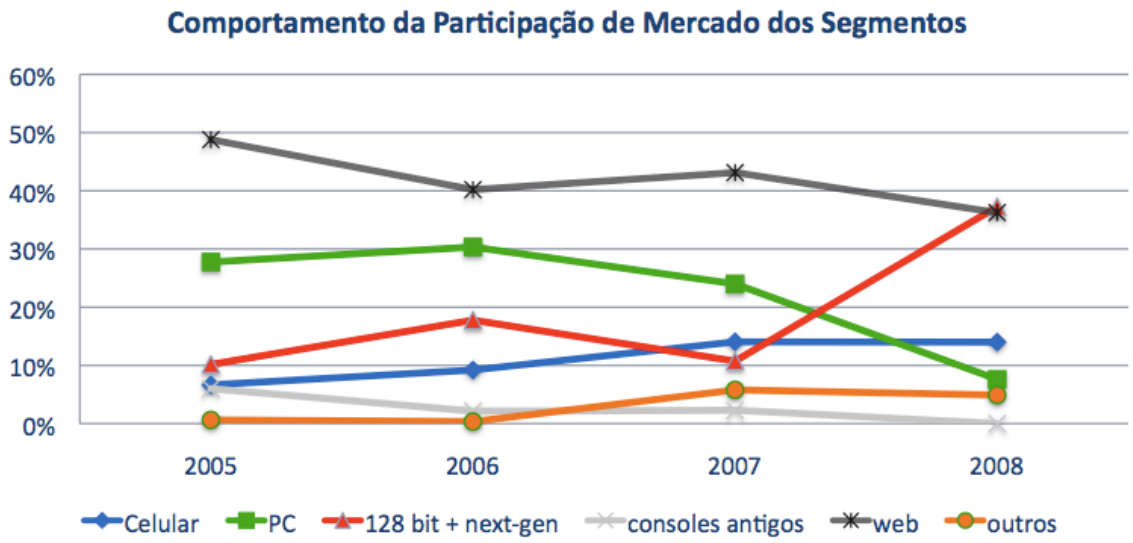


Figura 2 – Comportamento da Participação do Mercado nos Segmentos

2.2 O que se esperar de uma plataforma de desenvolvimento?

Existem diversas plataformas de desenvolvimento com a possibilidade de disponibilizar conteúdo para a web, muitas com a capacidade de manipular gráficos suficiente para permitir o desenvolvimento de jogos. Mas não é apenas esse parâmetro que viabiliza o seu uso para a implementação de jogos ou conteúdo interativo, existem diversas variáveis a serem consideradas. Em uma visão mais ampla existem dois pontos a serem analisados em cada plataforma, a **facilidade** no desenvolvimento e a **qualidade** do produto final. A seguir será apresentado as variáveis responsáveis por inferir a facilidade e a qualidade de uma plataforma.

- Facilidade: Refere-se a facilidade para o desenvolvimento de um jogo nas plataformas com recursos básicos necessários, esta subdivide-se em :

  - Reuso
    - Refere-se à capacidade de utilizar componentes prontos e reaproveitar blocos de códigos para o desenvolvimento dos jogos, aumentando assim a produtividade. Uma linguagem com suporte a

orientação a objetos ou que permite a criação de componentes gráficos possui uma grande capacidade de reuso.

- Curva de Aprendizado
  - Está associado a facilidade no desenvolvimento de aplicações simples no primeiro contato com a plataforma, analisando a dificuldade de realizar tarefas simples como desenhar elementos na tela por exemplo.

- Interação com outras ferramentas
  - Geralmente no desenvolvimento de um jogo são utilizadas diversas ferramentas para criar dos elementos de um jogo, sendo separadas essencialmente pelas ferramentas gráficas e de modelagem que são utilizadas pelos artistas gráficos e as ferramentas de programação que são utilizadas pelos programadores. Em uma plataforma que aceita o máximo de formatos de arquivos gerados em outros softwares ela possuirá uma alta capacidade de integração. Esta variável refere-se a capacidade da plataforma de interagir com outras ferramentas de desenvolvimento.

- Testes
  - Uma plataforma de desenvolvimento robusta possui diversas ferramentas de testes para que o desenvolvedor possa checar se o seu sistema esta executando o fluxo correto. Uma plataforma com essas características é fundamental para o desenvolvimento de jogos complexos, sendo uma variável essencial nesta análise.

- Disponibilização do Produto
  - A dificuldade no desenvolvimento de um jogo é de enorme relevância para uma análise, mas depois que o jogo está pronto um outro processo é necessário antes que os usuários possam utilizar os

jogos, este é chamado de *deploy*. O *deploy* é o processo de empacotar os binários do projeto e disponibilizar na web, esta variável analisa a complexidade desse processo em cada plataforma. Alem da capacidade da plataforma de poder rodar os seus jogos sem a necessidade do browser.

- Qualidade: Esta refere-se a impressão do usuário sobre o jogo desenvolvido, incluído aspectos gráficos e características de seu funcionamento. Esta subdivide em:

    - Gráficos
        - Refere-se às características das tecnologias na hora de apresentar os gráficos na tela. Analisando se a plataforma permite um pós processamento, ou seja, se é possível um efeito de blur por exemplo, ou se ela possui recursos de anti-aliasing na sua renderização.

    - Interatividade
        - A interatividade disponibilizando pela plataforma define muito a capacidade dos jogos que podem ser construídos com ela, por exemplo, se uma plataforma permite o uso de webcams ou microfones, um desenvolvedor pode se utilizar desses recursos para que seja uma entrada de dados de um jogo por exemplo.

    - Dificuldade da Instalação
        - Não adianta apenas que a tecnologia possua resultados excelentes nos jogos se para que o usuário possa utilizar-la antes seja necessário um complexo trabalho de instalação. Esta variável avalia dados como os pré-requisitos para a instalação do plug-in e a facilidade de atualizar para as versões mais novas.

    - Desempenho

- Ultima variável e não menos importante, esta refere-se ao desempenho da tecnologia nas máquinas, analisando dados como o tempo de espera para inicializar uma aplicação e a performance desta aplicação em máquinas com recursos medianos.

No próximo capítulo essas variáveis serão utilizadas para realizar o estudo comparativo baseado nas capacidades teóricas de cada plataforma.

3. Estado da Arte

Para a seleção das plataformas, inicialmente foram recolhidos as principais necessidades e requisitos para o desenvolvimento de um jogo com níveis avançado de interatividade.

Os pré-requisitos estabelecidos para os tecnologias são:

* Reproduzir som
* Capturar teclas do teclado e posição do Mouse
* Controle de Animações
* Rodar processos paralelos (*Threads*)
* Conexão com um ou mais servidores externo
* Rodar nos principais browser
* Rodar nos principais sistemas Operacionais
* Possuir uma API de renderização 2D

As plataformas devem ser capaz de realizar os pré-requisitos sem a necessidade de integração com uma tecnologia externa, desta forma será mais fácil realizar as comparações visto que as possibilidades de integração principalmente com as tecnologias web são praticamente infinitas. Desta forma ficará também mais fácil e estável desenvolver os jogos utilizando apenas um tipo de tecnologia, reduzindo a complexidade do sistema e reduzindo no custo de profissionais.

Segundo a Net Applications (Empresa que analisa e estuda as características das plataformas que acessam a internet, gerando estatísticas do uso das plataformas), o Windows XP, Vista e o MAC OS X 10.5 juntos corresponde a cerca de 92.27% dos sistemas operacionais conectados a internet, e o Internet Explorer, o Firefox e o Safari juntos correspondem a 97.1% dos browser utilizados mundialmente. Com base nesses dados foi definido como pré-requisitos das plataformas selecionadas, rodar o lado cliente

do jogo nos sistemas operacionais, Windows XP, Windows Vista e Mac OS X 10.5, rodando nos browsers Internet Explorer, Firefox e Safari [3].

Baseado nos pré-requisitos definidos foram encontradas três tecnologias de grande relevância para o desenvolvimento de jogos para a web, foram elas a plataforma Java da Sun, o Silverlight da Microsoft e o Flash da Adobe. Descreveremos a seguir o surgimento e as características de cada uma das tecnologia, focando no desenvolvimento dos jogos.

3.1 Java

A seguir será apresentado inicialmente como surgiu a plataforma Java e qual é o seu foco, mostrando em seguida uma análise dos recursos que a plataforma prover para o desenvolvedor.

3.1.1 História

A Plataforma Java e a linguagem começaram como um projeto interno da Sun Microsystems em Dezembro de 1990, sendo um projeto baseado nas duas linguagens de implementação mais amplamente utilizadas do mundo, C e C++. Isso imediatamente deu a Java uma enorme base de programadores altamente qualificados que estavam implementando a maioria dos novos sistemas operacionais, sistemas de comunicação, sistemas de base de dados, aplicativos de computador pessoal e softwares de sistemas do mundo.[12] O projeto tinha como objetivo desenvolver uma linguagem de programação que seja fácil de desenvolver, de identificar erros, que seja orientado a objeto e principalmente que seja multiplataforma. Baseado nesses princípios vieram as tecnologias de garbage collector que tira das mão do desenvolvedor a responsabilidade de controlar e remover as referências de memória inválida do software em tempo real, a máquina virtual que prove a característica de multiplataforma para a linguagem, permitindo que o mesmo código Java funcione em diversas plataformas com poucas ou nenhuma modificação, bastando que exista uma máquina virtual para o dispositivo, além de apresentar com mais clareza os erros ocorridos no software facilitando a sua correção.[6]

Em Junho de 1994, a plataforma foi redirecionada para a Web. Surgiu então a possibilidade de desenvolver conteúdo interativo e dinâmico em Java na web, alem de ser largamente utilizada para a implementação de softwares complexos para servidores [13].

3.1.2 Ambiente de Execução

Para se executar um software desenvolvido em Java necessitam-se de dois componentes, o máquina virtual, e um conjunto de bibliotecas que disponibiliza uma série de serviços para o programa. [6]

3.1.1.1 Máquina Virtual

O coração da plataforma Java é o conceito de um processador "virtual", que executa os programas formados por bytecodes Java. Este bytecode é o mesmo independente do hardware ou sistema operacional em que o programa será executado. A plataforma Java disponibiliza um interpretador, a JVM, que traduz em tempo de execução, o bytecode para cada instrução nativa do processador. Isto permite que uma mesma aplicação seja executada em qualquer plataforma computacional que possua uma implementação de máquina virtual. [6]

3.1.1.2 Biblioteca de Classes

Nos sistema operacionais modernos, um módulo formado por códigos de grande reuso e organizado e disponibilizado para simplificar o trabalho do desenvolvedor. Este código encontra-se, normalmente, na forma de bibliotecas dinâmicas que a aplicação utiliza durante execução. Como a plataforma Java é independente do sistema operacional as aplicações que utilizam Java não pode depender dessas bibliotecas, por isso, a plataforma Java disponibiliza um grande conjunto padronizado de bibliotecas de classes, que contém praticamente o mesmo número de funções encontradas nos sistemas operacionais [6].

3.1.1.3 Multiplataforma

Uma das principais vantagens do Java é o enorme uso da comunidade da característica multiplataforma do Java. Praticamente todos os sistemas operacionais modernos possuem uma implementação da máquina virtual do Java, permitindo que uma vez implementado um sistema Java sem dependências do sistema operacional, esta pode ser portado para qualquer sistema que possua a versão mínima do Java necessário para executá-lo. Existindo também máquinas virtuais portáteis que permitem a execução de códigos Java também em dispositivos móveis e sistemas embarcados [12].

3.1.3 Linguagem de Programação

A linguagem utilizada pela plataforma Java é uma linguagem de programação que recebe o mesmo nome da tecnologia. Esta foi criada para ser utilizada na plataforma Java, sendo compilada em bytecodes pelo desenvolvedor para finalmente ser interpretada pela JVM em tempo de execução na máquina do cliente. Existem atualmente diversas maneiras de desenvolver em Java, podendo integrar várias linguagens ao projeto, como próprio Java, orientada a objetos e fortemente tipada, e as linguagens interpretadas como o Groovy, o JRuby e o Jython, que são linguagens de script, com tipos dinâmicos e que são transformadas em bytecodes em tempo real, reduzindo performance do software mas permitindo um desenvolvimento rápido com tipos dinâmicos [6].

3.1.4 Desenvolvendo na Plataforma

Para desenvolver em Java é necessário um kit chamado Java Development Kit (JDK) que é disponibilizado pela Sun. Este kit contem o Compilador Java e um importante conjunto de bibliotecas e ferramentas para o desenvolvimento, alem de ser acompanhado pelo Java Runtime Environment (JRE). O kit é livre e open source e pode ser baixado no site da Sun [6].

Alem do kit de desenvolvimento existem um conjunto de IDEs – Integrated Development Environment, software que reúne diversas características e ferramentas de apoio ao desenvolvedor de software com o objetivo de agilizar este processo [6] - free e open source para o desenvolvimento de aplicações Java. Um dos destaques de IDE para Java é a IDE Eclipse, baseada em plug-ins essa IDE inicialmente tinha foco em Java mas hoje em dia existem plug-ins para diversas linguagens.

Logo, para o desenvolvedor é possível desenvolver uma aplicação Java sem nenhum custo em ferramentas, pois as já existem ferramentas excelentes e gratuitas no mercado e também não é necessário pagar royalties para a Sun para utilizar a sua tecnologia Java sendo uma excelente ferramenta para a produção de jogos.

3.1.4.1 Java na Web

Atualmente existem duas maneiras de disponibilizar uma aplicação Java na Web, a primeira e a mais antiga é utilizando a tecnologia Applet e a segunda é utilizando a Web Start, descreveremos as características de cada uma a seguir.

3.1.4.1.1 Java Applet

Primeiramente vamos explicar o que é applet. Apple é um componente de software que roda dentro de um contexto de outro programa, por exemplo, quando você está navegando no site do youtube, o vídeo que esta sendo apresentado é renderizado por uma tecnologia independente do browser, e o restante do site é renderizado pelo browser em si, ou seja, é uma aplicação independente que complementa a aplicação principal. [6]

O Java Applet foi a primeira tecnologia Java a sair do ambiente de execução do desktop e ir para a web. Foi introduzida na versão da linguagem Java em 1995 com o objetivo de prover mais interatividade a uma pagina web, complementado as limitações do HTML. Um applet Java pode ser executado dentro de um browser que der suporte a JVM ou por um AppletViewer – uma aplicação independente do browser para testar um applet.

Graças a independência da plataforma dos bytecodes do Java, os Java applets podem ser executados em diversas plataformas, como o Windows, Unix, Mac e o Linux [6][12].

3.1.4.1.2 Web Start

O Java Web Start surgiu em 2001, diferente dos Java applets, esta tecnologia permite que o usuário execute um programa online sem a necessidade do browser, utilizando uma janela do sistema operacional como se fosse um aplicação nativa, esta forma de executar tem uma enorme vantagem em comparação com os applets, pois alguns problemas de compatibilidade do Java com o browser como é encontrado com os applets simplesmente não existe.Alem da tecnologia web star possuir uma configuração de sandbox com menos restrições, permitindo aplicações mais complexas. Sua desvantagem é que o web start não consegue se comunicar com o browser tão facilmente como é feito com os applets [6][12].

O Web Start se utiliza da tecnologia JNLP (Java Network Lauching Protocol) para descrever como a aplicação Java funcionará. A descrição fica armazenada em um XML, onde este contem informações como a localização dos pacotes jar e qual a classe principal da aplicação, alem de poder ser inserido parâmetros para a aplicação. Uma vez executado o JNLP, o Java ira iniciar o download para a máquina do usuário dos pacotes jar necessários para o funcionamento de aplicação, logo apos executando a aplicação em uma janela nativa do sistema operacional, caso a aplicação necessite de permissões mais avançadas do sistema o Java ira perguntar para o usuário se ele permite essa interação.

3.1.5 Java para Jogos

A plataforma Java permite um conteúdo interativo e rico em mídias permitindo rodar diretamente no browser, sendo assim uma excelente alternativa para o desenvolvimento de jogos para o browser. O Java permite uma manipulação avançada com gráficos na tela, possuindo uma integração com o OpenGL, o que permite o desenvolvimento de jogos tridimensionais e também possui uma API avançada de

manipulação de som e vídeos. O Java também consegue acessar dispositivos da máquina do usuário como webcam e microfone, podendo ser uma excelente entrada de dados de um jogo. Porem dependendo da complexidade do jogo, o tempo de desenvolvimento pode ser muito elevado, pois para acessar periféricos com o Java é necessário acessar muitas classes, carregando muitos objetos para o projeto. Uma solução seria utilizar bibliotecas prontas próprias para desenvolver jogos, abstraindo a complexidade de acessar periféricos e deixando fácil para o desenvolvedor o uso das mesmas.

3.2 Silverlight

Neste tópico será apresentado como surgiu a mais nova plataforma da Microsoft para o desenvolvimento de aplicações web, apresentando posteriormente os requisitos para a sua execução, as linguagens de programação suportadas e por fim as ferramentas existentes para o desenvolvimento na plataforma.

3.2.1 História

O Silverlight foi realmente apresentando na MIX conference em 2007. Antes do evento o silverlight era conhecido apenas como uma tecnologia base para a interface chamada WPF/E – Windows Presentation Foundation / Everywhere. A primeira versão do silverlight não possuía CLR – *Common Language Runtime*, linguagem de compilação intermediária que serve para agilizar a interpretação da linguagem em tempo real, sendo a base para uma tecnologia multiplataforma [14]. Ela foi possuía suporte a apenas um pequeno subconjunto de XAML, linguagem de marcação para a descrição da interface, permitindo construir aplicações completamente em XAML ou um misto entre XAML e Java Script. Como ela não possuía CLR, não existia o passo de compilação, passando toda a interpretação do Java script para o cliente [7]. As vantagens iniciais do silverlight foram [7]:

- Layout Básico – O Canvas é apenas um componente de layout, deixando os elementos de interface com o usuário sendo posicionada de forma absoluta.

- Controles Básicos – Os componentes possuem comandos de controle e captura da entrada de usuários, facilitando a checagem de eventos. Esses componentes ganharam mais recursos com a versão do silverlight 2.0.
- Gráficos 2D – Com uma geometria baseada em classes, que permitia maior flexibilidade, mais não permite que um componente seja diretamente inserido a interface com o usuário, e forma baseada em classes, que permite colocar elementos diretamente na interface, provendo a capacidade de desenhar formas 2D.

- Mídias – Com suporte a mídias como imagens e vídeos, permitindo facilmente o download das mídias, otimizando o uso da banda de internet.

- Animações – Prove a capacidade de executar animações configuradas direto no WPF.

- Pinceis e Transformações – Permite desenhar com pinceis com imagens de textura, vídeo, cores e cores com gradiente.

Para funcionar o silverlight necessita de um plug-in instalado no lado cliente, bastando que o cliente possua a versão mais nova do plug-in instalada para que todas as aplicações desenvolvidas com a tecnologia silverlight funcione, visto que a Microsoft garante a retro compatibilidade de sua plataforma.

O silverlight ganhou mais relevância com o lançamento de sua versão 2.0. Esta adicionou comandos mais completos para os seus componentes e grande performance com a compilação do silverlight para um código intermediário, compilando as linguagens da plataforma .Net, alem do Java script, permitindo a interpretação do código intermediário para a linguagem de máquina mais rapidamente. Foi adicionado também um melhor suporte para a rede, incluindo a capacidade de se comunicar via socket [14].

3.2.1 Ambiente de Execução

Neste tópico será apresentado os requisitos necessários para executar uma aplicação desenvolvida na plataforma .NET.

3.2.1.1 Máquina Virtual

Para rodar uma aplicação desenvolvida em silverlight o usuários antes precisa ter a máquina virtual .NET instalada. Também chamada de .NET *runtime*, esta é responsável pela interpretação em tempo real dos *assemblies* .NET, seu funcionamento será explicado no próximo sub-tópico.

A plataforma .Net da Microsoft é a base do ambiente de desenvolvimento para Silverlight e vários outros produtos Microsoft. Podendo ser instalada em qualquer sistema operacional da Microsoft, esta inclui um enorme conjunto de bibliotecas para o desenvolvimento na plataforma, alem da máquina virtual que permite a execução de todos os softwares desenvolvidos na plataforma. [7]

3.2.1.2 Biblioteca de Classes

A base de classes inclusas pela plataforma contem uma série de bibliotecas prontas para o desenvolvimento de vários tipos de software, incluindo para o gerenciamento de interface, manipulação e conexão com base de dados, criptografia, comunicação pela rede e algoritmos [7]. Uma vez utilizando bibliotecas externas, esta é compilada junto com o projeto, e será agora uma dependência para a execução do mesmo, necessitando sempre a presença dos seus binários juntos com os binários da aplicação principal.

3.2.2 Linguagem de Programação

Para a manipulação de interface, a plataforma Net trabalha com uma linguagem de marcação de layout, o XAML. Esta permite que o desenvolvedor desenhe interfaces

baseada em tags, separando a modelagem da interface da lógica do programa. Desta forma é possível desenvolver apenas o layout completo do software para depois escrever toda a sua lógica. Esta arquitetura serve tanto para desenvolver aplicações para desktop da plataforma .Net como para as aplicações Silverlight. Apos concluído o desenvolvimento da aplicação o código é compilado para assembly .Net e é comprimido junto com a linguagem de descrição XAML usando a algoritmo zip e armazenado em uma extenssão .xap, descreveremos esse processo mais detalhadamente a seguir [8][9].

3.2.1.1 *Common Language Infrastructure* (*CLI*)

O *Common Language Infrastructure* é uma especificação desenvolvida pela Microsoft que descreve a execução do código e o ambiente de execução para uma séries de plataformas, incluído a plataforma .Net, o mono e o *portable* .Net. A especificação define um ambiente que prove uma linguagem neutra da plataforma para o desenvolvimento e execução, incluindo funções, captura de exceções, *garbagecollector,* segurança e interoperabilidade [9][10].

Abaixo segue um diagrama de como o CLI interage com as diversas linguagens de programação da plataforma .NET:

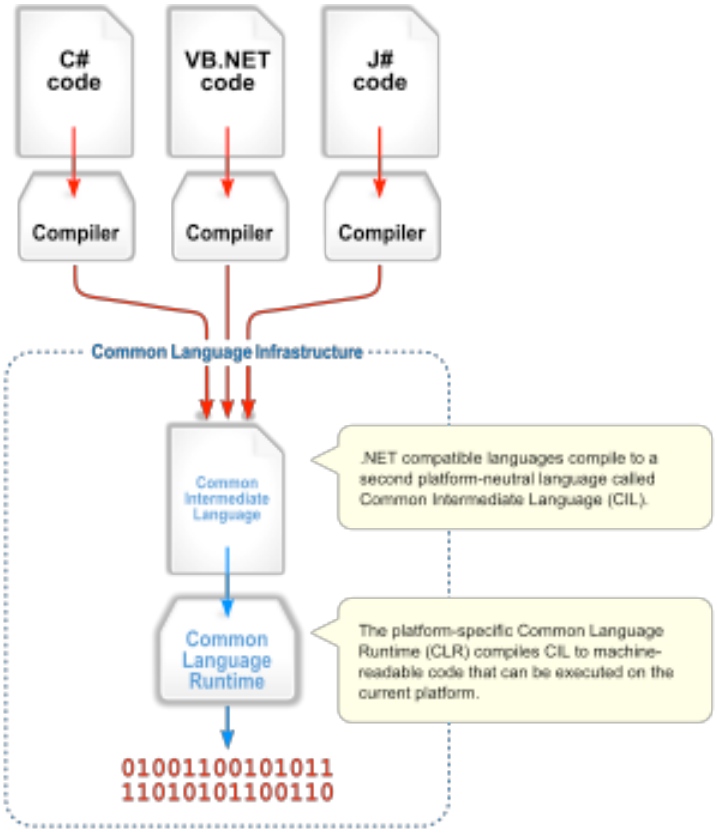


Figura 3 – Arquitetura do CLI

3.2.1.2 .NET *Assemblies*

Na plataforma .NET o código assembly da plataforma é parcialmente compilado pelas bibliotecas da plataforma para poder ser disponibilizado. O código assemblyé armazenado no formato PortableExecution (PE), comum entre as plataformas Windows para todos os DLLs e EXEs. Essa estratégia de código parcialmente compilado e a interpretação desses assemblies em tempo de execução é semelhante a arquitetura do Java, onde os códigos Java são compilados para os assemblies do JVM e depois interpretado para código de máquina em tempo real [8][9][14].

3.2.1.3 Segurança

A plataforma .NET possui o seu próprio mecanismo de segurança, este possui duas principais características, o *Code Access Security* (CAS), e a validação e verificação. [9]

O CAS é associado ao código assembly, ele permite ou recusa o uso de recursos da máquina dependendo da origem do código, dependendo se ele foi executado da máquina local ou se foi baixado e executado da internet ou intranet, bloqueando ou permitindo as execuções de bibliotecas específicas, caso algum assembly tente acessar algum recurso que não está permitido, e caso ocorra uma exceção de segurança é lançada.[9]

O segundo passo é a verificação e validação. Quando o assembly é carregado a máquina virtual realiza diversos teste antes de executá-lo, verificando a consistência dos metadados e tabelas da linguagem, e dos arquivo de descrição da interface XAML. Mas isso não garante a consistência completa do programa, necessitando ter a validação que verifica se o código é seguro e verificando os acessos a módulos do sistema. O algoritmo é bastante conservador podendo chegar a conclusão de que uma aplicação segura é insegura. Códigos definidos como inseguros podem ser executados se na validação do assembly o passo de verificação for desprezado, geralmente só se despreza esse tipo de assembly se este for executado em uma máquina local [9].

A plataforma também possui um conceito semelhante ao conceito do Java, de isolar os processos que estão sendo executados no framework .NET. Este conceito é chamado de Appdomains, que permite que um código seja carregado e descarregado independentemente dos outros Appdomains. Desta forma uma aplicação complexa fica mais tolerante a falhas, permitindo que em caso de crash do sistema apenas um módulo do sistema seja afetado sem afetar os outros módulos. Os Appdomains podem ter privilégios diferentes, permitindo que uma aplicação isole um módulo potencialmente inseguro, permitindo a implementação de um sistema baseado em plug-ins por exemplo [9].

3.2.1.4 Bibliotecas de Classe

Os conjuntos de classes disponíveis pela plataforma são organizadas em hierarquia de *namespaces*. As classes disponíveis contem um enorme numero de funções que amplia a capacidade das aplicações desenvolvidas na plataforma .NET, permitindo acesso ao sistema de arquivos do sistema operacional e podendo realizar tarefas como leitura e escrita de arquivos no sistema, renderização gráfica de forma rápida, interação com banco de dados locais e remotos, leitura e manipulação de XML, algoritmos básicos como manipulação de arvore binária, lista encadeada e muitas outras [7][9].

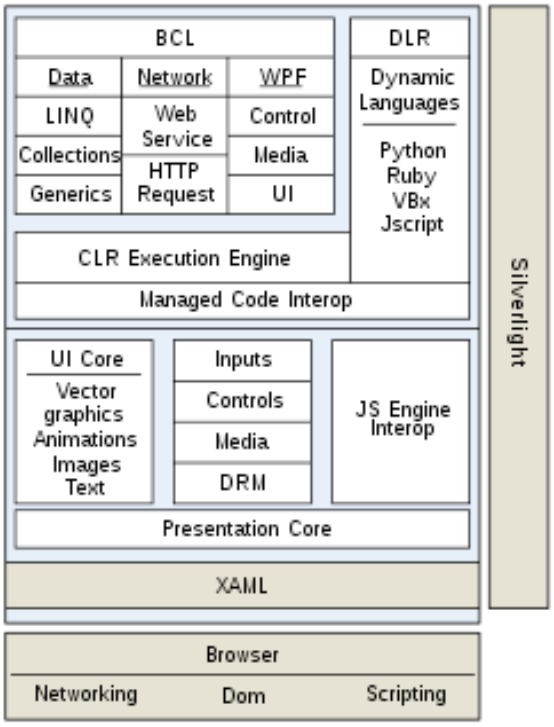


Figura 4 – Arquitetura

3.2.3 Ferramentas de Desenvolvimento

Aplicações silverlight podem ser escritas em qualquer linguagem de programação da plataforma .NET. Quanto a plataforma de desenvolvimento a Microsoft criou uma plataforma própria para o silverlight chamada Microsoft ExpresssionBlend, esta possui

foco no desenvolvimento na criação de interfaces e animações, permitindo também a escrita de código. Mas para projetos grandes e complexos o mais aconselhado é trabalhar a parte de codificação no Visual Studio com o plug-in SilverlightTools instalado. Este permite que o desenvolvedor possa testar acompanhar o passo a passo do programa em tempo de execução da aplicação em um processo chamado de *debugger*.

3.3 Flash

A tecnologia flash esta presente em cerca de 99% dos computadores conectados a internet segundo uma pesquisa realizada pela Adobe em 97 [16]. Estando na frente de tecnologias bem mais antigas como o Java por exemplo, a seguir apresentaremos como surgiu esta tecnologia, como é o desenvolvimento nesta plataforma e qual o futuro que a Abobe esta direcionando esta tecnologia dominante nos browsers.

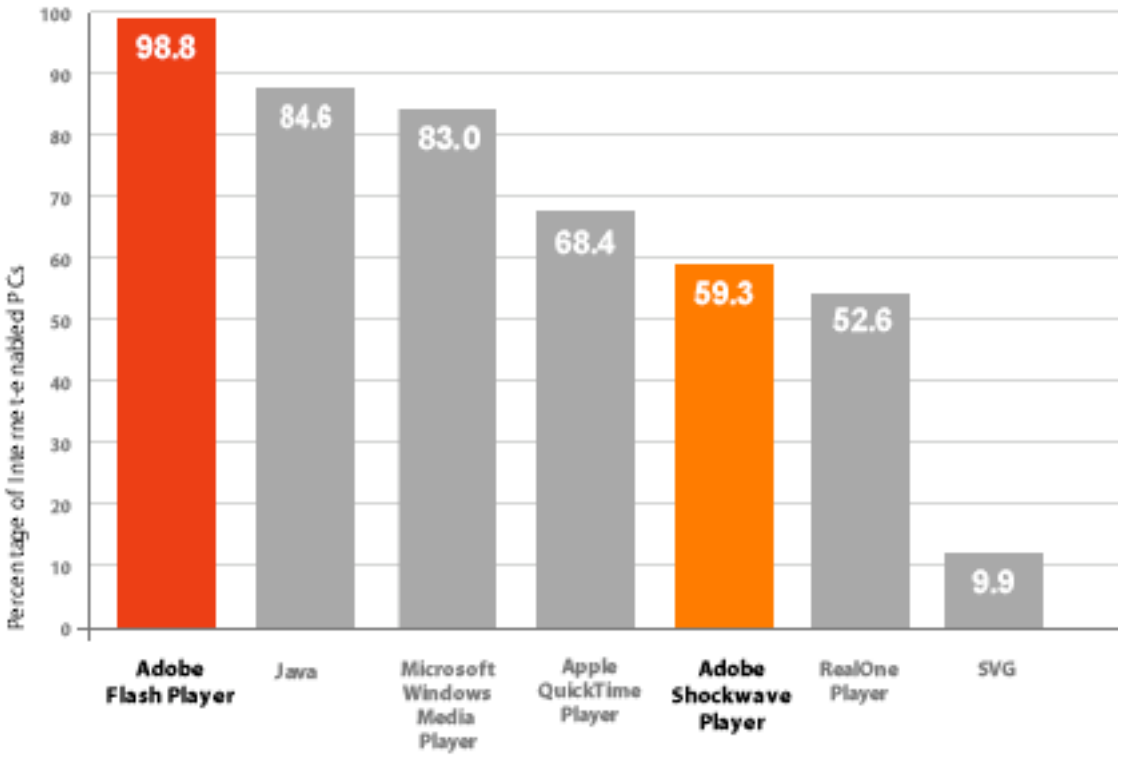


Figura 5 – Percentagem de plug-ins instalados nas máquina conectadas a internet

3.3.1 História

O Flash nasceu em 1996, pertencendo inicialmente a Macromedia, inicialmente era uma ferramenta para a criação de multimídia para a web. Com o objetivo inicial de ser uma ferramenta para fazer animações baseada em vetores, logo tornou-se bastante popular na web, inicialmente foi implementado um script básico chamado Actionscript que servia apenas para a manipulação das animações, com o tempo o actionscript foi ganhado mais características de linguagens e ganhado a capacidade de fazer interações cada vez mais complexas. Adquirida pela adobe em 2005 o flash já era popular entre os sites interativos, sendo a tecnologia dominante para o uso de animações em web sites [6].

3.3.2 Ambiente de Execução

O Adobe Flash é multiplataforma, trabalhando de forma semelhante as outras duas tecnologias anteriormente apresentada, o Java e o Silverlight, compilando a linguagem em um formato intermediário, este é interpretado em tempo real pelo flash player para a linguagem de máquina, podendo ser executado localmente na máquina do usuário em uma máquina virtual chamada flash player *standalone*, no browser através do plug-in do flash ou até em dispositivos móbiles com o flash lite (flash player mobile), este com recursos mais limitados [10].

Uma das principais vantagens do flash é que o seu código intermediário fica todo armazenado em um único arquivo, junto com todas as animações, mídias e gráficos que não são acessados de um repositório externo. Este arquivo de extensão swf,, se a aplicação foi desenvolvida sem dependência externa como imagens ou web services, pode facilmente ser portado para qualquer máquina ou servidor, bastando apenas que a máquina onde será executado possua o flash player igual ou superior a da versão que a aplicação foi desenvolvida.

3.3.3 Linguagem de Programação

A linguagem de programação para a plataforma flash é o ActionScript, que atualmente encontra-se na versão 3.0. Esta possui o maior foco para a manipulação dos componentes gráficos do flash, mas também possui suporte para codificação mais complexas, iremos falar a seguir sobre as diversas versões do ActionScript.

O ActionScript nasceu com o simples objetivo de gerenciar as animações em flash, controlando a seqüência das animações e permitindo a escrita de scripts básicos, com pouca interatividade e com capacidade bastante limitada [6].

Apos a necessidade dos usuários de poder interagir mais com os elementos gráficos do flash, foi introduzida a implementação ECMAScript na linguagem, deixando a sintaxe do ActionScript semelhante ao Java Script, mas com objetivos e bibliotecas distintas. O ActionScript ganha então a capacidade para realizar tarefas com o conteúdo mais interativo, permitindo a manipulação de componentes como botões, campos de textos e menus [6][10].

A partir da versão Flash MX 2008, foi introduzido o ActionsScript 2.0, esta possuía um foco muito alem do controle de animações, permitindo a escrita de códigos com lógica de programação mais complexas. Com suporte a orientação a objetos, o ActionScript agora permitia a escrita de softwares um pouco mais complexos, sendo assim mais fácil o seu entendimento e escrita [6].

A partir da versão do player do flash 9, uma nova versão do script foi lançada, o ActionScript 3.0, com um verdadeiro suporte a orientação a objetos e com um conceito renovado de lançamento e captura de eventos, o ActionScript agora possuía um clareza de código muito superior as outras versões, com uma lógica de desenvolvimento muito semelhante à linguagens já bem difundidas como o Java, o ActionScript agora deixava de ser uma plataforma de desenvolvimento apenas para desenvolver aplicações interativas e se transformava em uma excelente solução para o desenvolvimento de aplicações gerenciais de grande porte. Sendo mais robusto e estruturado esse novo ActionScript tornou-se a

linguagem padrão da Abobe para as diversas plataformas de desenvolvimento, como o Flash, o Flex e o Adobe Air [6][10][11].

A plataforma Flex Possui um funcionamento muito semelhante com a plataforma Silverlight da Microsoft, possuindo uma linguagem para o a criação de interface baseada em XML, o MXML e a linguagem de script para o desenvolvimento da lógica do programa, deixando os dois conceitos separados e facilitando o reuso de código com essa arquitetura. O XAML permite que a interface seja prototipada sem a necessidade de escrever uma linha de código, deixando o desenvolvimento mais descentralizado e permitindo que apos a definição da interface o desenvolvedor possa escrever a lógica do programa separadamente.

3.3.4 Desenvolvimento na Plataforma

Atualmente o Flash possui duas plataformas desenvolvidas pela Adobe para o desenvolvimento de aplicações na tecnologia. A primeira chamasse na sua atual versão, o Flash CS4, esta possui o foco na criação e manipulação de interfaces, com ferramentas gráficas para vetores bastante robustas, porem apesar de possibilitar a escrita de códigos na plataforma o Flash CS4, esta possui uma IDE fraca para a programação, com um suporte fraco a testes e debugger. Logo utilizar somente esta plataforma pra o desenvolvimento de aplicações é aconselhado apenas para projetos pequenos com poucas linhas de código. A segunda plataforma de desenvolvimento criada pela adobe chama-se Flex Builder, esta baseada na plataforma Open Source Eclipse para o desenvolvimento da sua IDE possui o foco para o desenvolvimento de códigos, com um excelente suporte para testes e *debugger* o Flex é a solução ideal para o desenvolvimento de códigos complexos e projetos com muita integração de tecnologia, e com uso de web services.

As duas plataformas da Adobe podem trabalhar em conjunto permitindo que o artista gráfico desenvolva as suas interfaces na plataforma gráfica Flash CS4 e posteriormente possa exportar as interfaces no formato compilado SWF ou no formato de componente SWC, o desenvolvedor então carrega esses arquivos na plataforma Flex

Builder como recursos do projeto. Desta forma o código fica separado da interface, aumentando o reuso de código e melhorando a modularidade do sistema. A primeira solução de exportar as interfaces no modo SWF possui como principal vantagem a possibilidade do desenvolvedor poder carregar as interfaces de forma gradativa na medida que o usuário vai navegando no sistema, só que pelo fato dos arquivos do projeto compilado serem vários SWFs separados e dependentes entre si, isso dificulta o processo de porte do sistema de um lugar para outro. A outra solução de exportar a interface como um componente SWC tem como principal vantagem o fato da interface ficar inserida no SWF final, junto com o código fonte do sistema, sendo mais fácil o seu porte.

As duas plataformas apresentadas não são as únicas soluções para o desenvolvimento na tecnologia Flash, graças à Adobe ter criado um projeto Open Source do compilador Flash em Maio de 2008 várias soluções surgiram no mercado. Das soluções comerciais a que mais se destaca é o FDT, baseada no Eclipse se assemelha bastante ao Flex Builder, e possui versões nas duas principais plataformas, o Windows e o Mac OS. E das soluções *free* e *open source* a melhor solução é o FlashDevelop, esta com suporte apenas para Windows tornou-se uma excelente solução para o desenvolvimento na plataforma Flash sem nenhum custo.

3.3.4.1 Flash Open Source

Em 2007 a Adobe tornou o projeto Flex em Open Source, disponibilizando os códigos das APIs do Flash e do compilador desenvolvido em Java com a licença *Mozilla Public License*, permitindo que a comunidade possa participar no desenvolvimento das API´s futuras, possibilitando o porte desta plataforma para os mais diversos sistemas operacionais. Atualmente é possível desenvolver em flash nos Windows, no MAC OS, no Linux e no Solaris de forma gratuita e com todo o apoio da comunidade.

Agora com a nova plataforma Apollo é Possível desenvolver não somente aplicações para a web, mais também aplicações para o *desktop*, possibilitando que o desenvolvedor desenvolva e compile apenas uma versão da aplicação e esta possa ser

executada em qualquer sistema operacional com a máquina virtual local flash, chamada de *Adobe Air Player*.

3.4 Estudo Comparativo

Baseado nas variáveis definidas no capítulo anterior, o estudo comparativo foi realizado com base na **facilidade** e na **qualidade** de cada plataforma.

A análise a seguir foi realizada com os dados disponibilizados nos sites dos desenvolvedores de cada tecnologia. Apresentando apenas se a tecnologia dão suporte ou não à característica analisada. No próximo capítulo teremos os resultados das análises realizadas para este trabalho.

3.4.1 Facilidade

Neste tópico cada variável que compõe a classificação de facilidade de uma ferramenta será comentada para cada tecnologia e por fim será apresentado uma tabela com apresentando que possui a melhor alternativa para a característica em foco.

3.4.1.1 Reuso

- **Plataforma Java:** O Java é uma linguagem orientada a objetos e possui um conceito de empacotamento de API´s em arquivos JAR, este metodo melhora bastante o reuso de código. Possuindo uma enorme capacidade de expansão.

- **Plataforma Silverlight:** O Silverlight trabalha com as linguagens do pacote .NET, dentre as linguagens existes temos a linguagem C# que é a mais padronizada, com orientação a objetos e fortemente tipada, alem de possuir uma excelente capacidade de reuso de código. A plataforma .NET permite o

empacotamento de suas API´s externas em DLL´s, permitindo um reuso ainda maior de seu código.

- **Plataforma Flash:** Em sua versão atual do seu script o ActionScript 3.0 possui também um excelente suporte a orientação a objetos. Alem de possuir também um formato de empacotamento de suas classes para a criação de API´s externas. As classes são armazenadas em um formato chamado SWC, esta podem conter tanto as classes de códigos como podem conter conteúdo como imagens por exemplo.

3.4.1.2 Curva de Aprendizado

- **Plataforma Java:** O Java possui uma curva de aprendizado relativamente grande, parte disso vem do fato do Java possuir uma orientação a objetos muito complexa e fragmentada, com muitas classes e heranças dificultando o aprendizado. Por exemplo, não é trivial implementar uma simples leitura de um arquivo de texto em Java, é necessário a utilização de várias classes intermediárias para realizar tal feito, alem de possuir uma API complexa para o tratamento de interface gráfica, com a necessidade de importar várias classes para desenvolver tarefas simples.

- **Plataforma Silverlight:** O Silverlight possui um script de fácil entendimento, e possui excelentes referencias online. Porem existem pequenas complicações como por exemplo, para poder acessar os eventos do teclado é necessário que o código possua acesso ao elemento gráfico raiz por exemplo, dificultando um pouco a criação de códigos simples.

- **Plataforma Flash:** Dentre todas as plataformas apresentadas o Flash é a que possui a curva de aprendizado menor, permitindo que em poucas horas com a ferramenta, o desenvolvedor já consiga manipular eventos de tecla, acesso gráfico e manipulação de som, com um script poderoso é possível realizar

complexas animações com filtro escrevendo poucas linhas de código, para realizar o mesmo feito em Java seria necessário utilizar APIs externas complicadíssimas e escrever muitas linhas de códigos. Uma das grandes complicações do flash é o seu poder de trabalhar com vários eventos assíncronos, em parte essa característica é muito boa para o desenvolvedor, podendo realizar tarefas enquanto aguarda a finalização de outras, o problema é que dependendo da quantidade de eventos simultâneos pode ficar complicado para o desenvolvedor consegui administrar todas as possibilidades de ocorrência dos eventos.

3.4.1.3 Interação com outras ferramentas

- **Plataforma Java:** Por ser Open Source o Java já possui integração com diversos formatos de arquivos, permitindo que seja carregado por exemplo uma infinidade de formatos de imagens, áudio e vídeos com seus codecs específicos. O problema do Java é que essas bibliotecas estão espalhadas pela web, descentralizadas e muitas vezes não compatíveis entre si, dificultando para o desenvolvedor encontrar as bibliotecas para carregar um formato de arquivo que seja muito específico.

- **Plataforma Silverlight:** O silverlight possui apenas um suporte completo para JPEG e PNG, possuindo também suporte limitado para outros formatos. Deixando a integração com ferramentas gráficas muito limitadas.

- **Plataforma Flash:** O flash consegue importar praticamente todos os formatos de imagens disponíveis, alem de possuir uma excelente integração com outras plataformas da Adobe como as ferramentas gráficas Photoshop e Fireworks por exemplo. Essa integração das ferramentas facilitam na hora de portar os gráficos desenhados no Fireworks por exemplo para serem utilizados em um Jogo, deixando o flash na posição da ferramenta que

possui a melhor integração com outras ferramentas gráficas para o desenvolvimento de games.

3.4.1.4 Testes

- **Plataforma Java:** As ferramentas de desenvolvimento mais utilizadas no mercado na plataforma Java possuem recursos de Testes e Debuggers excelentes, são elas o Eclipse e o Netbeans. Ambos com suporte a JUnit e com Debuggers embutido, realizar testes nesses frameworks é bem fácil e eficiente.

- **Plataforma Silverlight:** Para o desenvolvimento de códigos complexos a Microsoft aconselha a utilizar a IDE Visual Studio, esta possui uma das melhores ferramentas de Debugger do mercado, com um complexo sistema de edição de variáveis em tempo de execução deixando a etapa de testes bem simples para o desenvolvedor.

- **Plataforma Flash:** A ferramenta mais robusta recomendada pela Adobe para o desenvolvimento na plataforma Flash é o Flex Builder, este baseado na IDE Eclipse possui a mesma ferramenta de Debugger disponibilizada para o desenvolvimento na plataforma Java, com suporte a JUnit o Flash também possui excelentes recursos para teste.

3.4.1.5 Disponibilização do Produto

- **Plataforma Java:** Para se disponibilizar um software Java utilizando a API gráfica Applet é bem simples, bastando comprimir os binários do projeto para o formato .jar criar uma tag no HTML dizendo que o conteúdo é um Applet para que o Browser passe o processo para a máquina virtual Java. Uma vantagem desse processo é que os binários necessários ficam todos armazenados em um só arquivo, dependendo da implementação podendo

conter dependência externa ou não. Um Applet Java pode ser executado localmente na máquina do usuários, sem a necessidade do browser, bastando ter o um player de Applet Java instalado.

- **Plataforma Silverlight:** O Silverlight trabalha cada componente de forma separada, necessitando que cada componente seja disponibilizando separadamente. Logo é necessário colocar no servidor todos os conjuntos de classes e recursos da aplicação, necessitando que ao portar a aplicação de um lugar para outro seja necessário carregar todo o conjunto de dependências junto. Outra desvantagem do Silverlight é que não é possível executar um projeto desenvolvido nesta tecnologia localmente na máquina, sendo necessário sempre o browser para executar a aplicação. Uma vantagem neste modo de disponibilizar o conteúdo é que com os recursos do projeto ficam disponíveis para acesso, é possível indexar as imagens e os textos das aplicações Silverlight, diferentemente das aplicações Flash e Java que o conteúdo fica comprimido.

- **Plataforma Flash:** Semelhante ao Java, o Flash também comprime todos os códigos compilados e recursos como as imagens e animações em um único arquivo no formato .swf. Para carregar este arquivo basta adicionar uma tag HTML dizendo para o brower que existe uma aplicação flash, e passa o controle para o Flash Player. Outra vantagem do flash é a possibilidade de executar um arquivo swf localmente na máquina do usuário, sem a necessidade do browser. Sendo muito fácil disponibilizar e distribuir uma aplicação flash.

Segue abaixo o resultado das comparações entres as tecnologias no ponto de vista da facilidade. Os adjetivos que irão caracterizar os dados inferidos serão representados por: Excelente, Médio e Fraco.

Variáveis	Java	Silverlight	Flash
Reuso	Excelente	Excelente	Excelente
Curva de Aprendizado	Médio	Excelente	Excelente
Integração com outras ferramentas	Médio	Médio	Excelente
Testes	Excelente	Excelente	Excelente
Disponibilização do Produto	Excelente	Médio	Excelente

Tabela 1 – Estudo comparativo da produtividade em cada plataforma

3.4.2 Qualidade

Neste tópico serão comparadas as características de cada plataforma no que influenciam na qualidade do produto final, e por sim será apresentado uma tabela comparativa.

3.4.2.1 Gráficos

- **Plataforma Java:** Os gráficos do Java são baseado em Bitmap, sem recursos de Anti-aliasing nas bibliotecas gráficas padrões os gráficos da plataforma apresenta uma qualidade bem inferior às outras duas plataformas, deixando as aplicações com gráficos de baixa qualidade. É possível aplicar filtro sobre as imagens, mais para isso é necessário procurara bibliotecas capaz de realizar esses calculo, pois não existe em sua biblioteca padrão.

- **Plataforma Silverlight:** O silverlight trabalha com vetores para a renderização de seus gráficos, deixando as aplicações com uma aparência mais atraente, porem ainda não possui os recursos de filtros como filtro de *blur* ou *glow* por exemplo. Mas a Microsoft está prometendo esses recursos na próxima versão..

- **Plataforma Flash:** Os gráficos do Flash são baseados em Vetores, apesar de possuir bibliotecas para a manipulação de pixels, o foco do flash são os gráficos vetorizados. O flash desde a versão 8 possui também a capacidade

de aplicar filtros sobre os gráficos e vetores, possibilitando a criação de efeitos e gráficos de grande impacto visual, sendo a ferramenta com melhores resultados gráficos para a criação de conteúdos interativos para a web.

3.4.2.2 Interatividade

- **Plataforma Java:** Esta plataforma permite a interação do usuários com vários dispositivos externos, como webcam ou microfone, permitindo que sejam utilizados para a comunicação durante o jogo, ou que sejam até dispositivos de entrada de dados, como o controle para movimentar o personagem no jogo por exemplo.

- **Plataforma Silverlight:** Na sua versão atual não possui interação com dispositivos como webcam e microfone, mas a Microsoft está prometendo esses recursos na próxima versão.

- **Plataforma Flash:** Possui a recepção e tratamento de dados de forma fácil, permitindo que o usuário utilize a webcam e o microfone para interagir com a aplicação, com um sistema de permissão semelhante ao de Java, onde é solicitado permissão do usuário para utilizar tais recursos.

3.4.2.3 Dificuldade na Instalação

- **Plataforma Java:** Nos sistemas operacionais mais modernos a máquina virtual Java já vem instalada por padrão, possuindo versões para quase todos os sistemas operacionais da atualidade. Uma vez instalado o Java fica verificando na rede se possui uma versão superior disponível para ser atualizado, em alguns sistema operacionais as atualizações do Java são gerenciadas pelo próprio gerenciador de atualizações do sistema.

- **Plataforma Silverlight:** O silverlight ainda é uma tecnologia muito recente, por isso ainda não existem muitas máquinas com a tecnologia instala. Nos sites que são necessários o plug-in geralmente eles indicam o site para download caso o usuário não possua o *player*. Nos sistema operacionais da Microsoft, o player é instalado pelo próprio gerenciados de atualizações, e nas outras plataformas é necessário instalar de forma manual, necessitando ter acesso de administrador na máquina.

- **Plataforma Flash:** Segundo a Adobe o plug-in do Flash está presente em cerca de 98% das máquinas conectadas a internet, sendo a tecnologia dominante dos plug-ins web. Por ser uma tecnologia antiga e bastante difundida, é quase certo que o usuário irá possuir uma versão do flash player instalada no seu browser, até por ser utilizada em portais de grande acesso como o youtube por exemplo. Em alguns sistemas operacionais é necessário ser administrador para instalar o plug-in do Flash, mas existem browsers portáteis que já possuem o plug-in embutido, sem a necessidade ter o nível de administrador na máquina.

3.4.2.4 Desempenho

- **Plataforma Java:** Apos o browser identifica que o HTML possui uma aplicação Java, este inicia a máquina virtual caso ela não esteja iniciada e finalmente a aplicação Java começa a funcionar O problema é que o tempo entre iniciar a máquina virtual Java e a aplicação Java estar rodando é relativamente alto. Apesar desta impressão de lentidão, o Java consegue alocar mais recursos que os outros plug-ins, conseguindo renderizar aplicações tridimensionais por exemplo.

- **Plataforma Silverlight:** Quando uma aplicação Silverlight é encontrada no browser, esta é iniciada quase instantaneamente, passando a impressão de velocidade da plataforma. Já no ponto de vista da própria aplicação em si, ela

consegue dar respostas rápidas para aplicações simples e complexas dependendo da modularização do sistema, sendo uma excelente alternativa para aplicações web.

- **Plataforma Flash:** Semelhante ao Silverlight a inicialização do Flash Player é quase instantânea, alem de possuir excelentes respostas na execução de aplicações com muitos gráficos e animações, possuindo performance semelhante a da solução da Microsoft.

Segue abaixo a tabela com as comparações qualitativas das variáveis referentes a qualidade de cada tecnologia. Os adjetivos que irão caracterizar os dados inferidos serão representados por: Excelente, Médio e Fraco.

Variáveis	Java	Silverlight	Flash
Gráficos	Fraco	Excelente	Excelente
Interatividade	Excelente	Fraco	Excelente
Dificuldade de Instalação	Excelente	Médio	Excelente
Desempenho	Médio	Excelente	Excelente

Tabela 2 – Estudo comparativo da qualidade de cada plataforma.

4. Estudo de Caso

Dentre todas as plataformas apresentadas no capítulo anterior foram selecionadas duas delas, onde foi desenvolvido o mesmo jogo nas duas plataformas com a finalidade de comparar a complexidade de seu desenvolvimento em cada uma das tecnologias.

As plataformas selecionadas foram o flash por estar presente em 98% dos computadores conectados a internet [16] e ser praticamente a tecnologia dominante para o desenvolvimento de jogos casuais na internet e o Silverlight por ser uma recente alternativa ao flash, que vem ampliando o seu mercado em ritmo acelerado e recebendo atualizações de novos recursos com uma freqüência maior que a sua concorrente, alem de possuírem características semelhantes, possibilitando assim um estudo comparativo mais eficaz.

Para o desenvolvimento dos jogos foram utilizados utilizadas as ferramentas de desenvolvimento recomendada pelas produtoras de cada tecnologia. Para o desenvolvimento do jogo no flash foi utilizado o Flex Builder 3 da Adobe, e para o desenvolvimento na plataforma Silverlight foi utilizado o Visual Web Develop 2008 Express Edition da Microsoft.

O game design do jogo desenvolvido foi planejado em conseguir adquirir o máximo de variáveis possíveis para o estudo comparativo e ao mesmo tempo possuir uma história simples para não desperdiçar tempo em tarefas de pouca relevância para o estudo comparativo. Com base nisso o jogo não possui uma história, consiste em um puzzle onde o jogador possui uma canhão e o seu objetivo é apenas acertar o alvo de cada cenário. A complexidade consiste na idéia de utilizada uma física bidimensional no jogo, onde cada fase irá possuir várias estruturas físicas que dificultarão o acesso da bala ao alvo, necessitando que o usuários destrua as estruturas de forma planejada para não dificultar ainda mais o acesso ao alvo. A adição de uma engine física em cada plataforma permite um excelente teste de performance, e usos de recurso da máquina do usuário.

4.1 Estudo Comparativo

Neste tópico será apresentado os teste comparativos entre cada plataforma no ponto de vista prático, ou seja, dados comprovados durante o desenvolvimento dos jogos em cada plataforma, mas antes da realização dos testes comparativos será apresentado alguns fatores externos que influenciarão nos resultados das plataformas.

4.1.1 Fatores Externos

Alguns fatores externos devem ser considerados que podem deixar os dados constatados tendenciosos para uma plataforma ou para outra.

Inicialmente a minha experiência prévia em cada tecnologia. Sou desenvolvedor Flash ActionScript 3.0 à aproximadamente um ano e meio, e estou atualmente trabalhando com a tecnologia, diferentemente da tecnologia Silverlight onde tive o meu primeiro contato com graças a este trabalho. Possuo uma experiência de desenvolvimento em C# de aproximadamente 2 meses, o que me ajudou bastante no entendimento da plataforma Silverlight.

Outro ponto que deve ser considerado é que as engines físicas que foram inseridas em cada projeto são de desenvolvedores diferentes, possuindo um guia de uso diferente e com otimização distintas para cada plataforma, influenciando diretamente na quantidade de linhas de código escritos em cada plataforma para utilizá-las e influenciando na performance.

4.1.2 Resultados do Estudo

Para a realização dos estudos comparativos práticos entre as plataformas foram escolhidos inicialmente as variáveis que foram analisadas no estudo comparativo do capítulo anterior para serem comprovadas na prática.

Para a comparação no ponto de vista da **produtividade** foram escolhidas as seguintes variáveis:

- **Reuso:** A variável de reuso de código será testada analisando a quantidade de linhas de código necessária para o desenvolvimento do projeto.

- **Curva de Aprendizado:** A curva de aprendizado será inferida pelo tempo de desenvolvimento em cada plataforma, considerando a experiência prévia do desenvolvedor em cada plataforma.

- **Disponibilização do Produto:** Esta variável será inferida pela quantidade de arquivos gerados necessários para o funcionamento do jogo.

E para a comparação no ponto de vista da **qualidade** foi escolhida a seguinte variável:

- **Desempenho:** Está será inferida pelo uso de recursos da máquina a medida que será adicionado mais elementos físicos em cada plataforma, a fim de testa a capacidade de cálculos de cada plataforma.

Os resultados de cada teste comparativo serão apresentados a seguir:

4.1.2.1 Reuso

Como já foi citado anteriormente esta variável de reuso possui uma margem de erro por conta da engine física utilizada em cada plataforma, de modo a afetar a quantidade de linhas de código totais do projeto por conta das linhas necessárias para inicializar e manipular cada elemento na engine.

- Plataforma Flash: O flash com o ActionScript 3.0 possui uma excelente orientação a objetos, e por conta disso possui uma excelente capacidade de reuso de código, alem de apresentar uma API simples para a criação de threads o que permite a implementação de códigos paralelos de uma forma

bem simples. No desenvolvimento do jogo, a quantidade de linhas de código total do primeiro protótipo ficou em 673 linhas.

- Plataforma Silverlight: Semelhante ao flash o Silverlight também apresenta um excelente suporte a orientação a objetos e também uma API simples de ser manipulada. Porem para a captura de teclas do teclado é necessário ter uma atenção especial pois só é possível ter acesso as teclas através do elemento gráfico principal. Quanto a quantidade de linhas de código, o primeiro protótipo ficou com 1759 linhas, uma quantidade muito superior a quantidade de linhas do projeto em flash, uma das principais razões de enorme quantidade de linhas vem do fato de que as classes básicas da engine física para o flash já implementa de forma estruturada os modelos físicos primários, coisa que foi necessário implementar em Silverlight. Por causa dessa diferença refiz a contagem de linha de código desconsiderando esse módulo do sistema, e nessa contagem o projeto ficou com 1391 linhas de código, o que mostra que independente da engine física é necessário escrever mais para implementar algo no Silverlight do que no Flash.

Abaixo segue a arquitetura de classes do jogo em cada plataforma.

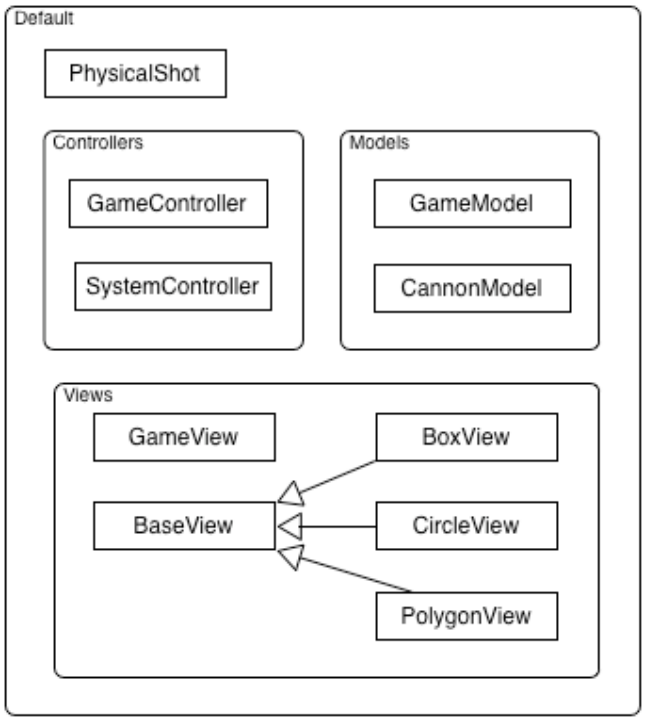


Figura 6 – Diagrama de Classes do Jogo implementado em Flash

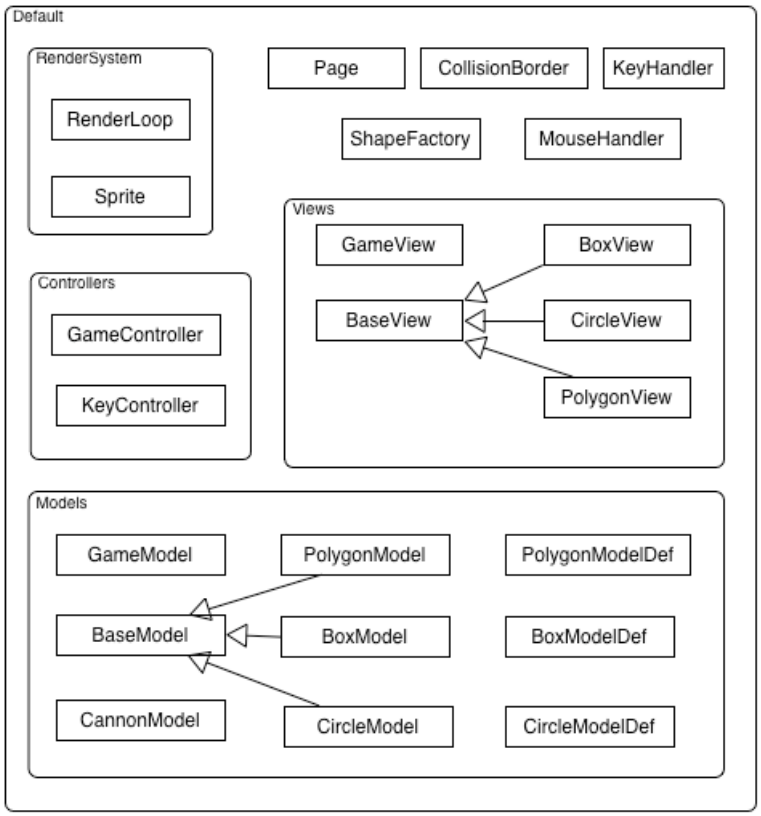


Figura 7 – Diagrama de Classes do Jogo implementado em Silverlight

4.1.2.2 Curva de Aprendizado

Esta variável possui um fator externo como já foi citado anteriormente que afeta diretamente os resultados dessa variável. Sou desenvolvedor Flash a quatro anos e possuo experiência em ActionScript 3.0 a um ano e meio, diferentemente do Silverlight onde tive o meu primeiro contato com esse projeto.

- Plataforma Flash: O primeiro protótipo do jogo em flash foi desenvolvido em apenas dois dias. O fator principal desse curto tempo de desenvolvimento vem por conta da grande experiência na tecnologia.

- Plataforma Silverlight: Já para o desenvolvimento em Silverlight necessitou de 5 dias de desenvolvimento. Onde o primeiro dia foi para aprender a utilizar a plataforma, sendo um tempo bem curto graças a excelente base de referências disponibilizadas pela Microsoft e por possuir conceitos bem semelhantes aos existentes no Flash, sendo bem fácil migrar o conhecimento de uma plataforma para a outra. O restante dos dias foram gastos na aprendizagem e implementação dos elementos físicos da plataforma. O tempo gasto na plataforma em si foi bem semelhante ao tempo de desenvolvimento no flash.

4.1.2.3 Disponibilização do produto

- Plataforma Flash: O Flash compila todo o projeto em apenas um único arquivo, o swf, sendo bastante fácil de portar, alem de ser possível executá-lo localmente como já foi dito anteriormente.

- Plataforma Silverlight: Diferente do Flash o Silverlight necessitou de quatro arquivos para o seu funcionamento. Foram duas DLLs, uma da engine física e uma do projeto PhysicalShot, um arquivo XAP que é chamado pelo HTML para executar o aplicativo Silverlight e um arquivo de Manifest com as

permissões da aplicação. A ausência de qualquer um desse arquivos impede a execução do projeto, alem de só poder ser executado no browser, como já foi dito anteriormente.

4.1.2.4 Desempenho

O desempenho das plataformas serão inferidos pelo custo de memória e pelo uso do processador de vários browsers rodando apenas a aplicação em teste. Não será possível testar a quantidade de frames por segundo na plataforma Silverlight porque a engine física FarseerPhysics possui uma implementação que impede que a taxa de frames por segundo do projeto seja reduzida, para isso quando a aplicação passa para a engine física calcular um número muito grande de objetos físicos, este calcula a posição real dos elementos mas atualizando os gráfico numa taxa baixa, logo ao invés de reduzir toda a taxa de atualização como é feito com a engine física Box2D do flash o FarseerPhysics apenas reduz a taxa de atualização dos gráficos, mantendo todo o resto do sistema com a taxa correta, o que impossibilita o cálculo de capturar da taxa de frames por segundo sem que seja necessário modificar a engine física.

Foram realizados uma bateria de testes de performance em diversos browsers e sistemas operacionais, segue abaixo as configurações das maquinas utilizadas nos testes.

Dados da máquina 1.

- Sistema Operacional: Mac OS X 10.5.7
- Processador: Core 2 Duo 2.4Ghz
- Memória RAM: 4Gb
- Placa Gráfica: Intel X3100

Dados da máquina 2

- Sistema Operacional: Windows XP Professional
- Processador: Core 2 Duo 2.4Ghz
- Memória RAM: 4Gb

- Placa Gráfica: Intel X3100

E as versões dos browsers utilizados.

Sistema Operacional MAC OS X:

- Firefox 3.5 RC
- Opera 10 b
- Safari 4.0.1

Sistema Operacional Windows XP

- Firefox 3.5 RC
- Opera 10 b
- Safari 4.0.1
- Internet Explorer 8
- Google Chrome

Os testes foram realizados dez vezes em cada browser de cada sistema operacional e foi utilizada a média aritmética dos dados adquiridos. Todos os browser utilizaram a mesma configuração: Apenas uma aba rodando a aplicação e browser recém aberto.

Seguem abaixo os resultados da bateria de testes realizados em cada uma das maquinas.

	Silverlight		Flash	
Browser	Processador	Memória	Processador	Memória
Firefox 3.5 RC	36%	105 Mb	32%	97 Mb
Opera 10 b	28%	51 Mb	19%	42Mb
Safari 4.0.1	25%	68 Mb	22%	56 Mb

Tabela 3 – Testes de Performance na Máquina 1

	Silverlight		Flash	
Browser	Processador	Memória	Processador	Memória
Firefox 3.5 RC	20%	55	25%	55
Opera 10 b	20%	56	20%	39
Safari 4.0.1	20%	61	22%	63
Internet Explorer 8	24%	60	26%	61

Tabela 4 – Testes de Performance na Máquina 2

De acordo com os estes realizados identificamos que o as tecnologias possuem resultados muito semelhantes, onde o Flash se mostrou mais eficaz rodando na plataforma MAC OS X, e o Silverlight se mostrou mais otimizado para a plataforma da Microsoft, o Windows XP. Mas visto que os resultados foram muito próximos, podemos considerar que as tecnologias possuem os mesmo desempenho rodando sobre a mesma estrutura. Considerando esta variável como um ponto não decisivo para a escolha da plataforma.

5. Considerações Finais

Neste capítulo descreveremos as conclusões obtidas e discutiremos quais características levou a tais resultados. Em seguida um tópico com sugestões de trabalhos complementares a este estudo.

5.1 Conclusão

No desenvolvimento de um jogo para a web é de extrema importância que a empresa ou o desenvolvedor faça uma análise profunda de qual tecnologia deve ser utilizada com base nas características do jogo a ser implementado.

Tendo em vista este cenário, as avaliações de custo benefício das plataformas analisadas referente a qualidade do produto e a facilidade de desenvolvimento mostraram os seguintes resultados:

Qualidade: Em relação a qualidade do produto final, o Flash demonstrou ser bastante superior ao Silverlight, apresentando um nível de interatividade com microfone e webcam que inexiste na tecnologia concorrente, alem de possuir a capacidade de manipular som. Outra vantagem do Flash para o usuário final é que ele poderá interagir com aplicações Flash em praticamente qualquer sistema operacional, pois diferente do Silverlight que apenas da suporte ao Windows e ao Mac OS X, o Flash também da suporte ao Linux.

Produtividade: Quanto a produtividade o Flash também se mostrou superior à solução da Microsoft, pois apresentou um excelente nível de integração com outras ferramentas, alem de possuir um modo de disponibilização de conteúdo mais simples, e como foi demonstrado nos testes prático é necessário escrever bem menos linhas de código no flash do que em Silverlight para desenvolver um jogo.

Logo, o Flash é a melhor solução para o desenvolvimento de jogos para a web, mas o Silverlight é uma plataforma muito nova, e que vem tento atualizações com mais freqüência que o Flash, sendo um enorme ameaça para a dominância do Flash no mercado de jogos casuais na web, sendo de grande relevância estudos comparativos para cada nova versão de cada tecnologia..

6.2 Trabalhos Futuros

Este trabalho possuiu foco para o estudo comparativo de tecnologias web para jogos causais, selecionando plataformas com apenas suporte a elementos gráficos bidimensionais. Atualmente na web existem duas grandes tecnologias com suporte a jogos tridimensionais rodando direto no browser, são elas a Unity3d e o Torque 3D com o InstantAction. Em um trabalho futuro essas novas tecnologias poderiam ser analisadas e comparadas.

Segundo a Microsoft, a versão do Silverlight 3.0 possuirá uma série de recursos que deixará mais próximo da sua concorrente Flash, necessitando um novo estudo comparativo quando esta nova versão for lançada oficialmente.

Referências

[1] 2008 Essential Facts about Computer and Video Game Industry.
Pesquisa publicada pela Entertainment Software Association (ESA). Ano:
2008.

[2] A indústria brasileira de jogos eletrônicos
Um mapeamento do crescimento do setor nos últimos 4 anos.
Pesquisa 2008, versão 1.0
Disponível em: http://www.abragames.org/docs/Abragames-Pesquisa2008.pdf Acesso em: Março / 2009

[3] Empresa responsável pela análise de uso da internet. Disponível em: http://marketshare.hitslink.com/

[4] Engine Física em Java. Disponível em http://www.cokeandcode.com/phys2d/ Acesso em: Maio / 2009

[5] Wikipédia – Enciclopédia Online. Java Plataform. Disponível em http://en.wikipedia.org/wiki/Java_Platform Acesso em Abril / 2009

[6] Wikipédia – Enciclopédia Online. Flash Plataforma. Disponível em http://en.wikipedia.org/wiki/Flash_Platform Acesso em Maio / 2009

[7] Wikipédia – Enciclopédia Online. Silverlight. Disponível em http://en.wikipedia.org/wiki/Silverlight Acesso em Maio / 2009

[8] Wikipédia – Enciclipédia Online. .NET Assembly. Disponível em http://en.wikipedia.org/wiki/.NET_assembly. Acesso em Maio / 2009

[9] Wikipédia – Enciclopédia Online. .NET Framework. Disponível em http://en.wikipedia.org/wiki/.NET_Framework. Acesso em Abril / 2009

[9] Wikipédia – Enciclopédia Online. Common Language Infrastructure. Disponível em http://en.wikipedia.org/wiki/Common_Language_Infrastructure Acesso em Maio / 2009

[10] Roger Braunstein, Mims H. Wright, Joshua J. Noble. ActionScript 3.0. Ano 2008

[11] Colin Moock. Essencial ActionScript 3.0, O'REILLY. Ano: 2008

[12] H. M. Deitel, P. J. Deitel. Java como Programar. Ano: 2005

[13] James Gosling, Bill Joy, Guy Steele, Gilad Bracha. The Java Language Especification – Second Edition 2009

[14] Jeff Scanlon. Accelerated Silverlight 2. Ano: 2008

[15] Praticidade do Flash, Adobe. Disponível em http://www.adobe.com/products/flash/features/?view=toptimesavers. Acesso em Março / 2009

[16] Dominância do Flash Player no mercado de Plug-ins. Disponível em http://www.adrianblog.com/web-development/flash-player-installed-on-987-percent-of-online-computers.html Acessado em Maio / 2009.

[17] Tabela Comparativa do Flash com o Silverlight. Disponível em http://silverlight.net/forums/t/3015.aspx Acessado em Maio / 2009

[18] Uma séria de blogs com comentários de desenvolvedores sobre cada as tecnologias Flahs e silverlight. Disponível em http://geekglue.blogspot.com/2007/04/silverlight-vs-flash.html. Acessados em Abril / 2009

[19] Um estudo comparativo entre o Flash e o Silverlight. Disponível em http://www.smashingmagazine.com/2009/05/09/flash-vs-silverlight-what-suits-your-needs-best/. Acessado em Abril / 2009

[20] Tutoriais para iniciantes em Silverlight. Disponível em http://silverlight.net/GetStarted/. Acessado em Maio / 2009

[21] Fórum com dicas de desenvolvimento em Silverlight. Disponível em http://silverlight.net/forums/18.aspx. Acessado em Maio / 2009

[22] Engine Física para Silverlight, FarseerPhysics. Disponível em http://farseerphysics.codeplex.com/. Acessados em Maio / 2009

[23] Engine Física para Flash, Box 2D. Disponível em http://box2dflash.sourceforge.net/. Acessado em Maio / 2009

[24] Tutoriais da engine física Box2D para flash. Disponível em http://box2dflash.org/node/7. Acessado em Maio / 2009

[25] IGDA – Casual Game White Paper. Disponível em http://www.igda.org/casual/IGDA_Casual_Games_White_Paper_2008.pdf Acessados em Março / 2009

[26] IGDA – Casual Games Quartely. Disponível em http://www.igda.org/casual/quarterly/igda_casual_game_quarterly_4_1.pdf. Acessado em Março / 2009

[27] IGDA – Web Downloadable Games White Paper. Disponível em http://www.igda.org/online/IGDA_WebDL_Whitepaper_2004.pdf. Acessado em Março / 2009

Apêndice A – Documento de Game Design

Physical Shot

Game Design
Versão 1.0

1.0 Histórico do Projeto

Esta seção tem com objetivo apresentar a evolução deste documento. É apresentado aqui o conteúdo da versão inicial do documento e, para cada versão seguinte, o que de novo foi acrescentado ou o que foi removido.Esta seção tem com objetivo apresentar a evolução deste documento. É apresentado aqui o conteúdo da versão inicial do documento e, para cada versão seguinte, o que de novo foi acrescentado ou o que foi removido.

Titulo do Documento		Plano de Projeto	
Equipe		Mário Barbosa <mbaj@cin.ufpe.br>	
Comentários			
HISTÒRICO DE REVISÔES			
Revisões	**Data**	**Descrição**	**Autor**
01	17/05/2009	Elaboração do Documento	Mário

2.0 Visão Geral do Jogo

2.1 Filosofia

2.1.1 Aonde Quero Chegar?

O objetivo deste jogo é realizar um teste comparativo das plataforma onde foram desenvolvidos, utilizando recursos avançados das plataformas.

2.1.2 Tipo do Jogo

O Physical Shot é um jogo bidimensional, onde o jogador controla um canhão com o objetivo de acertar o alvo de cada cenário.

2.1.3 Plataforma de Execução

O jogo será desenvolvido para ser executado nos browsers com suporte ao plug-in do Flash e ao plug-in do Silverlight

2.2 Questões Comum

2.2.1 O que é o Jogo?

O Physical Shot é um jogo de tiro bidimensional tradicional. O seu diferencial é a física implementada na engine do jogo, necessitando que o usuário se utilize dos recursos do cenário, como derrubar uma pilastra ou utilizar uma gangorra para conseguir deixar o alvo acessível.

2.2.2 Por que criar o jogo?

O principal foco do desenvolvimento deste jogo foi para realizar o estudo de caso do estudo comparativo entre as plataformas Flash e Silverlight. Alem da paixão por jogos influenciar na escolha deste tema.

2.2.3 Onde se ambienta o jogo?

O ambiente do jogo é um mundo onde as construções são feitas apenas com a sobreposição das peças, com sistemas mecânicos e verdadeiras estruturas que se utilizam da física para se manter firme.

2.2.4 O que se controla no jogo?

O jogador controla apenas o canhão, escolhendo a força da bala e o angulo apenas.

2.2.5 Quantos personagens se controla no jogo?

Apenas um único canhão disponível em todos os cenários.

2.2.6 Qual o foco principal do jogo?

O foco principal consiste em acertar os alvos de cada cenário se utilizando das estruturas físicas para deixá-los acessíveis.

2.2.7 O que o jogo apresenta de diferente?

O diferencial do jogo são as estruturas física que podem ser utilizadas a favor ou contra o jogador, precisando que o jogado pense e analise o cenário antes de dar um tiro e poder destruir a única chance de acertar o alvo.

3.0 Conjunto de Características

3.1 Características Gerais

- Jogo 2D com vista lateral
- Possui apenas um personagem que é o jogador
- Cenário com uma avançada física 2D implementada
- Dispositivo de entrada é apenas o teclado
- O jogo é destinado a apenas um jogador

3.2 Gameplay

- O Physical Shot possui apenas um único modo de jogo que é avançar entre os cenários até concluir todas as fases disponibilizadas no jogo.
- O jogador controla um canhão que muda o angulo e a força que a bala ira sair do tubo do canhão.
- O jogador deve atirar pensando em como livrar o caminho para deixar o alvo do cenário acessível.
- Inicialmente os alvos estarão com o acesso facilitado e com várias possibilidades de poder atingi-lo, na medida que o jogador vai passando de fases o jogo vai ficando cada vez mais complexo, dificultando o acesso da bala ao alvo e deixando poucas possibilidades do jogador poder passar de fase.

4.0 O mundo do jogo

4.1 Cenário

O jogo é dividido em vários cenários, inicialmente serão implementados apenas dois cenários para possibilitar o estudo comparativo. Cada cenários apresentará características que irão interferir na jogabilidade, as estruturas fundamentais do jogo são.

- Estruturas poligonais sobrepostas
- Correntes fixas nas paredes do cenário
- Motores com funcionamento circulares

4.2 Sistema de Renderização

O jogo é isométrico. Cada polígono é desenhado dinamicamente com as APIs gráficas de cada plataforma. O canhão e a bala serão Sprites importados a partir do formato png.

4.3 Game Engine

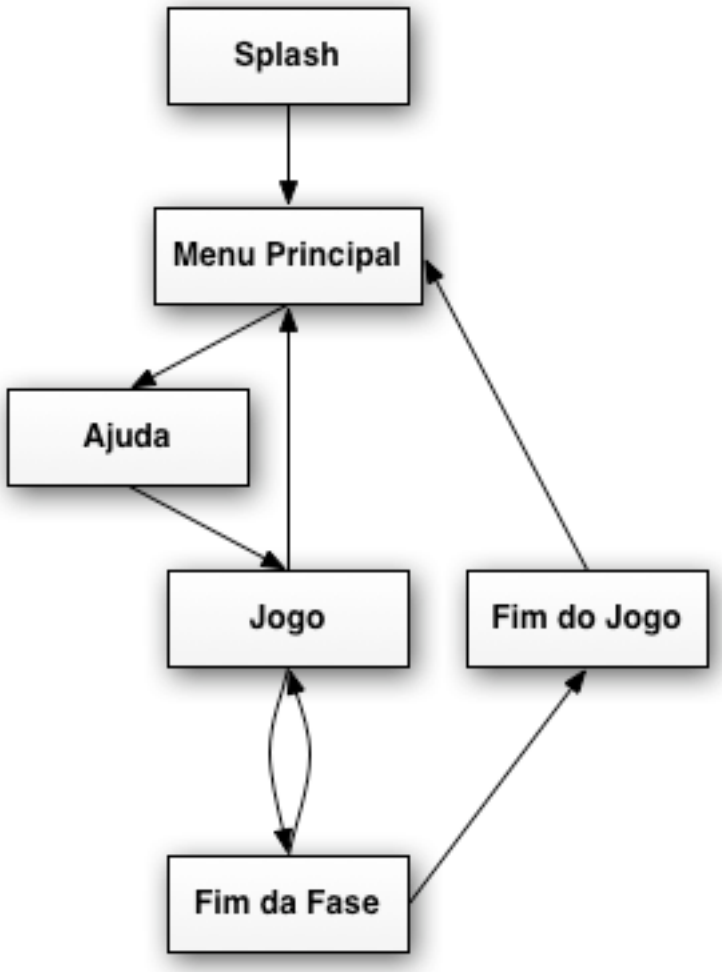
A engine do jogo tratará as seguintes tarefas:

- Tratamento de colisão: Engine física que tratará a colisão dos elementos físicos do cenários simulando a reação de cada tiro nos blocos e elementos do cenários. Caso seja detectado a colisão entre a bala e o alvo o jogo passa para a próxima fase iniciando a posição dos elementos para as posições predefinidas da fase corrente.
- Tratamento de Teclas: Toda a navegação do jogo serão realizados pelo teclado.
- Renderização Gráfica: Após cada loop da interação física os elementos físicos são representados por elementos gráficos e apresentados na tela.

5.0 Interface com o usuário

5.1 Visão Geral

A interface com o usuário do Physical Shot é constituída de menus navegáveis que permite ao usuário obter informações sobre o jogo, passar para a próxima fase e voltar mana o menu principal. Segue abaixo o diagrama de telas:



- **Splash:** Tela de apresentação do Jogo.
- **Menu Principal:** Tela de entrada do jogo com um botão que da acesso ao jogo.
- **Ajuda:** Tela com as instruções do jogo.

- **Jogo:** Tela do jogo em si.
- **Fim da Fase:** Quando o jogador consegue atingir o alvo esta tela de passagem para o próximo nível é apresentada, caso não existam mais níveis a tela de fim de jogo é apresentada.
- **Fim de Jogo:** Tela de fim de jogo, mostrando que o jogador conseguiu passar por todas as fazer.

5.2 Controles do jogo

As teclas de interação no jogo são as direcionais cima e baixo para mover mudar o angulo do canhão e a barra de espaço para atirar, quando mais tempo o jogador segurar a barra de espaço mais força o canhão transfere para a bala.

6.0 Jogo Single Player

6.1 Visão Geral

No modo Single Player existirá inicialmente duas fases, onde as fases seguintes só serão acessíveis depois de passar pela anterior. A dificuldade das fases serão progressivas, e com um gameplay diferenciado, necessitando que o jogador utilize uma linha de raciocínio diferente para conseguir passar de cada fase.

6.2 Horas de jogo

Na tua versão de teste com apenas duas fazer o tempo previsto de jogo deve chegar por volta dos 10 minutos, mas a previsão é de adicionar diversas fases desafiadoras para para atingir a média de uma hora de jogo.

6.3 Condições de vitória

A vitória de cada fase é atingir o alvo do cenários, ao conseguir atingir o alvo de todos os cenários o fim de jogo é alcançado.