



**UNIVERSIDADE FEDERAL DE SANTA CATARINA
CTC – CENTRO TECNOLÓGICO
INE – DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

ADOBE FLEX X GWT-EXT. Tecnologias RIA para aplicações Java com desenvolvimento no Eclipse.

Gustavo S. Schneider
Tiago Braun Corrêa

**Florianópolis
2009**



**UNIVERSIDADE FEDERAL DE SANTA CATARINA
CTC – CENTRO TECNOLÓGICO
INE – DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

***ADOBE FLEX X GWT-EXT. Tecnologias RIA para aplicações Java com
desenvolvimento no Eclipse.***

Trabalho de conclusão de curso apresentado ao
Curso de Sistemas de Informação como requisito
parcial à obtenção do grau de Bacharel em
Sistemas de Informação a Gustavo S. Schneider e
Tiago Braun Corrêa.

Orientador: Frank Siqueira.

**Florianópolis
2009**

*ADOBE FLEX X GWT-EXT. TECNOLOGIAS RIA PARA APLICAÇÕES JAVA
COM DESENVOLVIMENTO NO ECLIPSE.*

Trabalho de conclusão de curso submetido à Universidade Federal de Santa Catarina como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Frank Augusto Siqueira, Dr.

Universidade Federal de Santa Catarina

frank@inf.ufsc.br

Banca examinadora

Prof. Leandro José Komosinski

Universidade Federal de Santa Catarina

leandro@inf.ufsc.br

Prof. Patricia Della Mea Plentz

Universidade Federal de Santa Catarina

plentz@inf.ufsc.br

“Getting information off the Internet is like taking a drink from a fire hydrant.”
Mitchell Kapor

AGRADECIMENTOS

Aos nossos pais, familiares e namoradas, por estarem sempre ao nosso lado, em todos os momentos e por darem as oportunidades que nos trouxeram a este caminho.

Aos nossos professores, pelo tempo dedicado na busca de um ensino de qualidade, em especial, ao nosso orientador Frank Augusto Siqueira.

Pelo carinho de nossos amigos, sempre dispostos a nos ajudar.

Por toda estrutura disponibilizada pela UFSC.

RESUMO

Com o aumento da importância da Web como veículo de comunicação e o boom da Web 2.0 houve uma popularização de muitas tecnologias de interface rica (RIA), além da criação de novas.

Algumas delas não apenas deixam as aplicações mais modernas e bonitas, como também adicionam novas funcionalidades e facilidades, tanto para o usuário quanto para os desenvolvedores de software, que se vêm obrigados a adequar seus produtos a essas novas tecnologias.

Nesse contexto, surge a dúvida de qual tecnologia se adequa melhor a um determinado produto, haja vista que existem semelhanças entre as tecnologias, porém há também muitas características divergentes que diferenciam cada uma das tecnologias.

Neste trabalho são estudadas algumas dessas tecnologias e são comparados com mais detalhes o Adobe Flex e o GWT-EXT tendo como base o desenvolvimento de uma aplicação.

Tanto o desenvolvimento quanto a comparação tem por finalidade dar base a uma escolha da tecnologia mais indicada para desenvolvimento de interfaces ricas para WEB utilizando-se da linguagem Java tendo como IDE de apoio o Eclipse.

Palavras chave: Interfaces Ricas para a WEB, RIA (*Rich Internet Applications*), Adobe Flex, GWT (*Google Web Toolkit*), GWT-EXT, Java, Eclipse.

ABSTRACT

With the raising importance of the Web as a mean of communication and the boom of Web 2.0 there has been a popularization of many rich interfaces technologies and the creation of new ones.

Some of these not only make the application prettier and more modern, but also add more functionalities and facilities for the user as well as for the software developers. However, developers feel forced to adapt their products to these new technologies.

In this context one may question which technology is better for a certain product, as there are many similar characteristics among them, but also many distinct ones that differentiate one technology from another.

Using the development process as a base, this paper studies some of these technologies and thoroughly compares Adobe Flex and GWT-EXT.

Both the development and the comparison are intended to provide support for choosing the most appropriate technology when developing rich interfaces for WEB using the Java language and Eclipse as an IDE.

Keywords: Rich Interfaces for the WEB, RIA (Rich Internet Applications), Adobe Flex, GWT (Google Web Toolkit), GWT-EXT, Java, Eclipse.

LISTA DE FIGURAS

Figura 1 – Um painel com um <i>grid</i> editável onde um calendário dinâmico ajuda o usuário a escolher a data para o campo.....	13
Figura 2 – Utilizando Ext JS o código adiciona em um botão de id “mb8” a função que mostra uma caixa de alerta.	14
Figura 3 – Utilizando GWT-EXT o código cria um botão com o texto “Show Me” e adiciona nele a função que mostra uma caixa de alerta.	14
Figura 4 - Arquitetura da plataforma OpenLaszlo	16
Figura 5 - Produtos da linha Flex 2	18
Figura 6 - JavaFX Mobile nos PDAs	25
Figura 7 - Arquitetura JavaFX.....	28
Figura 8 – Exemplo de gráfico em linhas de visitasões.	35
Figura 9 – Exemplo de gráfico em pizza das fontes de tráfego.....	35
Figura 10 – Exemplo de gráfico em barras dos navegadores utilizados.	35
Figura 11 – Diagrama da interface da aplicação.	36
Figura 12 – Plataforma da aplicação	38
Figura 13 – Estrutura da aplicação, baseada no padrão J2EE	39
Figura 14 - Configuração dos módulos no application.xml	40
Figura 15 – web.xml: configuração do contexto WEB	40
Figura 16 – A classe ponto de entrada TccApp.java	43
Figura 17 – TccApp.gwt.xml – Configurações para a utilização da biblioteca GWT-EXT.	44
Figura 18 – Arquivos JavaScript da biblioteca GWT-EXT.	45
Figura 19 – Exemplo de layout complexo.....	46
Figura 20 – Tela inicial da aplicação, o portal no centro com seus <i>portlets</i>	47
Figura 21 - Gráfico de visitantes de um determinado domínio, por ano, mês, semana e dia.....	49
Figura 22 - Tela do módulo Adobe Flex	51
Figura 23 - Comparação entre a plataforma de desenvolvimento.	54
Figura 24 - Comparação entre a estrutura das tecnologias.....	55
Figura 25 - Comparação entre o suporte aos usuários.....	56
Figura 26 – Interface da classe de serviço que busca os dados do gráfico de visitantes no GWT.	57
Figura 27 – Código que faz a requisição dos dados do gráfico de visitantes no GWT.....	58

Figura 28 - Classe que trata o retorno de uma requisição no GWT.	58
Figura 29 – Código que declara como deve ser invocado o serviço que busca os dados do gráfico de visitas no Flex.	59
Figura 30 – Função que invoca o serviço no servidor que busca os dados do gráfico de visitantes no Flex.	59
Figura 31 – Funções no Flex que tratam o retorno do serviço ou possíveis erros ocorridos.	59
Figura 32 - Comparação entre código de programação.....	60
Figura 33 - Comparação entre desempenho das aplicações.	61
Figura 34 - Dificuldades encontradas no desenvolvimento.	62

LISTA DE TABELAS

Tabela 1 - Comparação entre tecnologia RIA, estudo comparativo.....	31
---	----

LISTA DE ABREVIATURAS E SIGLAS

RIA	Rich Internet Applications
PDA	Personal Digital Assistants
JEE	Java Enterprise Edition
JSE	Java Standard Edition
JME	Java Micro Edition
XML	Extensible Markup Language
XAML	Extensible Application Markup Language
OS	Operating System
OEM	Object Exchange Model
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
AS	ActionScript
AJAX	Asynchronous Javascript And XML
API	Application Programming Interface ou Interface de Programação de Aplicativos
JDBC	Java Database Connectivit
URL	Uniform Resource Locator ou Localizador de Recursos Universal
PC	Personal Computer ou Computador Pessoal
LGPL	Lesser General Public License
GUI	Graphical User Interface ou Interface Gráfica do Utilizador
JRE	Java Runtime Environment

SQL	Structured Query Language, ou Linguagem de Consulta Estruturada
IDE	Integrated Development Environment ou Ambiente Integrado de Desenvolvimento
AWT	Abstract Window Toolkit
SWF	Small Web Format
JDK	Java Development Kit
FLV	Flash Video
MP3	Abreviação de MPEG Moving Picture Experts Group
OSI	Open Source Initiative
IBM	International Business Machines Corporation
SOAP	Simple Object Access Protocol
ECMA	Ecmascript language specification
SDK	Software Development Kit
EJB	Enterprise Java Beans
GWT	Google Web Toolkit

SUMÁRIO

1	INTRODUÇÃO	4
1.1	CONTEXTUALIZAÇÃO DO PROBLEMA.....	4
1.2	PROPOSTA DE SOLUÇÃO	4
1.3	JUSTIFICATIVA	5
1.4	OBJETIVOS.....	5
1.4.1	Objetivo Geral	5
1.4.2	Objetivos Específicos.....	6
1.5	METODOLOGIA DE DESENVOLVIMENTO.....	6
1.6	ESTRUTURA DA MONOGRAFIA.....	6
2	FUNDAMENTAÇÃO TEÓRICA.....	8
2.1	SURGIMENTO DE RIA.....	8
3	TECNOLOGIAS RIAS.....	10
3.1	GOOGLE WEB TOOLKIT (GWT).....	10
3.2	EXT JS.....	12
3.3	GWT-EXT	14
3.4	LASZLO.....	15
3.4.1	LZX.....	16
3.5	ADOBE FLEX 2	16
3.5.1	MXML e ActionScript	18
3.6	MICROSOFT SILVERLIGHT	19
3.6.1	XAML.....	20
3.6.2	JavaScript.....	21
3.6.3	HTML.....	22
3.7	JAVAFX.....	23
3.7.1	JavaFX Mobile.....	24
3.7.2	JavaFX Script	25
3.7.3	Arquitetura JavaFX	27
3.8	COMPARAÇÃO.....	29
4	APLICAÇÃO.....	34

4.1	PROJETO DA APLICAÇÃO	34
4.1.1	Diagrama das Interfaces da Aplicação	36
4.2	DESENVOLVIMENTO DA APLICAÇÃO	37
4.2.1	Plataforma da Aplicação.....	37
4.2.2	Configuração da Aplicação.....	39
4.2.3	Desenvolvimento GWT-EXT.....	41
4.2.4	Desenvolvimento FLEX	48
5	CONCLUSÃO.....	52
5.1	PLATAFORMA	52
5.2	ESTRUTURA.....	54
5.3	SUORTE	55
5.4	CÓDIGO.....	56
5.5	DESEMPENHO	60
5.6	DIFICULDADES	61
5.7	CONSIDERAÇÕES FINAIS	62
5.8	SUGESTÕES PARA TRABALHOS FUTUROS.....	64
5.8.1	Estudo de caso.....	65
5.8.2	Ampliar o modelo da interface	65
5.8.3	Aplicar o modelo em outras tecnologias.....	65
6	REFERÊNCIAS	66
7	ANEXOS E APÊNDICES.....	68
7.1	ARTIGO	68
7.2	CÓDIGO FONTE DO MÓDULO GWT	80
7.2.1	Arquivo de configuração TccApp.gwt.xml	80
7.2.2	Arquivo de configuração jdoconfig.xml	81
7.2.3	Arquivo de configuração log4j.properties.....	81
7.3	CÓDIGO FONTE DO MÓDULO GWT (PACOTE CLIENTE)	82
7.3.1	Interface GwtConstants.java.....	82
7.3.2	Classe TccApp.java	82
7.3.3	Classe BrowsersChartDataCallback.java	82
7.3.4	Classe BrowsersDashboardWidget.java.....	83
7.3.5	Classe BrowsersWidget.java	84

7.3.6	Classe TrafficChartDataCallback.java	85
7.3.7	Classe TrafficDashboardWidget.java.....	86
7.3.8	Classe TrafficWidget.java	86
7.3.9	Classe VisitorsChartDataCallback.java.....	87
7.3.10	Classe VisitorsDashboardWidget.java	88
7.3.11	Classe VisitorsWidget.java	89
7.3.12	Classe BrowsersChart.java.....	89
7.3.13	Classe TrafficChart.java.....	90
7.3.14	Classe VisitorsChart.java	91
7.3.15	Classe Dashboard.java	92
7.3.16	Classe CenterPanel.java.....	94
7.3.17	Classe MainPanel.java	95
7.3.18	Classe MenuPanel.java	96
7.3.19	Classe TopPanel.java	98
7.3.20	Interface BrowsersChartDataService.java	98
7.3.21	Interface BrowsersChartDataServiceAsync.java	98
7.3.22	Interface TrafficChartDataService.java	99
7.3.23	Interface TrafficChartDataServiceAsync.java	99
7.3.24	Interface VisitorsChartDataService.java	99
7.3.25	Interface VisitorsChartDataServiceAsync.java.....	100
7.4	CÓDIGO FONTE DO MÓDULO GWT (PACOTE SERVIDOR)	100
7.4.1	Classe BrowsersChartDataServiceImpl.java.....	100
7.4.2	Classe TrafficChartDataServiceImpl.java	101
7.4.3	Classe VisitorsChartDataServiceImpl.java	102
7.5	CÓDIGO FONTE DO MÓDULO EJB.....	102
7.5.1	Classe BrowsersChartDataEjb.java	102
7.5.2	Interface BrowsersChartDataEjbLocal.java	104
7.5.3	Interface BrowsersChartDataEjbRemote.java	104
7.5.4	Classe TrafficChartDataEjb.java	105
7.5.5	Interface TrafficChartDataEjbLocal.java	106
7.5.6	Interface TrafficChartDataEjbRemote.java	107
7.5.7	Classe VisitorsChartDataEjb.java	107

7.5.8	Interface VisitorsChartDataEjbLocal.java	108
7.5.9	Interface VisitorsChartDataEjbRemote.java	109
7.6	CÓDIGO FONTE DO MÓDULO DE UTILITÁRIOS TCCAPP.JAR	109
7.6.1	Interface TccAppConstants.java	109
7.6.2	Classe XPathHelper.java	110
7.7	CÓDIGO FONTE DO MÓDULO FLEX (SERVIDOR)	111
7.7.1	Classe TCC_APPServlet.java	111
7.7.2	Classe XmlVisitantes.java	111
7.8	CÓDIGO FONTE DO MÓDULO FLEX (CLIENTE)	112
7.8.1	Interface de inicialização TCC_Application.mxml	112
7.8.2	Componente Flex Visitante_Componet.mxml	113
7.8.3	Componente Flex Navegadores_Components.mxml	113
7.8.4	Componente Flex Trafego_Component.mxml	114
7.8.5	Componente Flex DashBoardNavegador_Component.mxml	115
7.8.6	Componente Flex DashBoardTrafego_Component.mxml	115
7.8.7	Componente Flex DashBoardVisitante_Component.mxml	116
7.8.8	Arquivo ActionScript visitante.as	117
7.8.9	Arquivo ActionScript navegador.as	118
7.8.10	Arquivo ActionScript trafego.xml	118
7.9	CÓDIGO FONTE DA PASTA DADOS	119
7.9.1	Arquivo de dados dadosacessos.xml	119
7.9.2	Arquivo de dados dadosnavegadoresefontestrafego.xml	135
7.10	CÓDIGO FONTE DA PASTA DE CONFIGURAÇÃO META-INF	138
7.10.1	Arquivo de configuração application.xml	138
7.10.2	Arquivo de configuração jboss-app.xml	138
7.11	CÓDIGO FONTE DA PASTA DO CONTEXTO WEB WAR	138
7.11.1	Arquivo de requisição Flex requisicao.jsp	139
7.11.2	Arquivo de configuração appengine-web.xml	139
7.11.3	Arquivo de configuração logging.properties	139
7.11.4	Arquivo de configuração web.xml	140
7.11.5	Arquivo de interface inicial index.html	141
7.11.6	Arquivo de estilo TccApp.css	142

7.11.7	Arquivo de interface GWT-EXT TccApp.html.....	143
---------------	--	------------

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO DO PROBLEMA

Com o aumento da importância da internet como veículo de comunicação surge também a necessidade de melhorar a experiência do usuário na internet. Novas tecnologias de interface são desenvolvidas e utilizadas. Algumas delas não apenas deixam a interface mais moderna ou bonita, mas também adicionam novas funcionalidades e facilidades, tanto para o usuário quanto para o programador da mesma.

A importância da interface é enfatizada por Ana Carolina Costa Martins (2007), “Uma boa interface pode ser determinante na aceitação de determinado produto, uma vez que a execução das tarefas da aplicação depende da forma que os elementos são apresentados ao usuário”.

O *boom* da Web 2.0 levou à popularização de muitas tecnologias além da criação de novas, o que acaba gerando alguns receios para o programador que gostaria de atualizar a interface de seu sistema: qual é a tecnologia que deve ser adotada analisando desempenho da aplicação, suporte aos usuários, plataforma de desenvolvimento, dificuldades de implementação, código de programação e estrutura de projeto.

1.2 PROPOSTA DE SOLUÇÃO

Desenvolver um programa simples utilizando-se das tecnologias Adobe Flex e GWT-

EXT. Durante este desenvolvimento, anotar e realizar comparação entre as duas tecnologias nos quesitos levantados neste trabalho.

1.3 JUSTIFICATIVA

A necessidade de integrar ao sistema uma interface de ponta surge com a competitividade do mercado. Neste contexto, empresas de desenvolvimento de software vêm adotando novas tecnologias RIAs em busca da satisfação de seus clientes, sua permanência no mercado e a longo prazo um aumento na cadeia de clientes.

Porém a adoção de uma nova tecnologia interfere no ambiente da empresa tanto na estrutura operacional, tendo esta que redefinir seus processos de desenvolvimento de software, quanto no ambiente organizacional, com novas áreas, novas funções e contratações, além de treinamento.

Visto isso, a escolha da adoção de uma nova tecnologia deve ser estudada com afinco, tentando assim aumentar significativamente as chances de sucesso da organização.

1.4 OBJETIVOS

Podemos classificar os objetivos neste trabalho em gerais e específicos.

1.4.1 Objetivo Geral

O objetivo deste trabalho é comparar duas tecnologias RIAs promissoras, o Adobe Flex e GWT-EXT.

1.4.2 Objetivos Específicos

Segue objetivos específicos:

1. Comparar as tecnologias no quesito plataforma de desenvolvimento;
2. Equiparar as estruturas de projeto;
3. Examinar o suporte dado aos usuários;
4. Confrontar as tecnologias no quesito código de programação;
5. Verificar o desempenho das tecnologias;
6. Elencar as dificuldades de implementação encontradas;

1.5 METODOLOGIA DE DESENVOLVIMENTO

Este estudo será desenvolvido nas etapas descritas a seguir:

- 1 Conhecer as principais tecnologias;
- 2 Comparar as tecnologias abordadas;
- 3 Desenvolver uma aplicação exemplo simples com o Adobe Flex e GWT-EXT;
- 4 Comparar as tecnologias e os pontos levantados durante o desenvolvimento;

1.6 ESTRUTURA DA MONOGRAFIA

Este estudo está dividido em 5 capítulos. O capítulo 1 apresenta uma introdução ao tema, apresentando o contexto do mesmo, qual o problema engajado, e também as motivações do estudo.

O capítulo 2 traz um estudo teórico acerca de interfaces ricas para WEB.

O capítulo 3 aborda uma descrição de algumas das tecnologias RIAs existentes no mercado, realizando uma comparação entre estas.

No capítulo 4 desenvolvemos uma aplicação exemplo utilizando as tecnologias Adobe Flex e GWT-EXT .

No capítulo 5 apresentamos conclusões sobre as tecnologias e estudo realizado e também sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 SURGIMENTO DE RIA

O surgimento da Internet se deu em meados da guerra fria com fins militares, sem nunca passar pelas mentes daquela época a proporção que esta tecnologia iria tomar no mundo.

A rede mundial de computadores, ou Internet, surgiu em plena Guerra Fria. Criada com objetivos militares, seria uma das formas das forças armadas norte-americanas de manter as comunicações em caso de ataques inimigos que destruíssem os meios convencionais de telecomunicações. Nas décadas de 1970 e 1980, além de ser utilizada para fins militares, a Internet também foi um importante meio de comunicação acadêmico. Estudantes e professores universitários, principalmente dos EUA, trocavam idéias, mensagens e descobertas pelas linhas da rede mundial. (www.suapesquisa.com)

Hoje é quase impossível imaginar um mundo desconectado, desde lojas de pequeno varejo até indústrias de grande escalão estão totalmente entrelaçadas ao meio, não deixando dúvidas sobre o poder desta tecnologia. Contudo, seu crescimento e aprimoramento são necessários e sabidos.

Com o crescimento da internet, a distribuição e a manutenção de aplicações tornaram-se muito mais simples e rápidas. Utilizando um Browser, o usuário passou a acessar o último release sem ter que se preocupar com versionamento ou procedimentos de instalação. Mas um efeito colateral do paradigma web não tardou a aparecer. Interfaces mais pobres e menos intuitivas substituíram os sistemas “fat-client” antes largamente utilizados, forçando o usuário a se acostumar com o “clique esperar” das interfaces web. (MONTEIRO, 2008)

Segundo Doedelein a “incapacidade dos padrões Web (como HTML e CSS) para implementar funcionalidades de GUI mais avançadas, incluindo suporte rico a mídias em geral, o insuficiente respeito a estes padrões, mesmo quando estes deveriam ser suficientes e o fraco desempenho da linguagem JavaScript”, são os fatores principais para o surgimento das tecnologias RIA.

(...) O conceito de Rich Internet Application (RIA) é uma resposta a este problema. Permitindo que o usuário interaja com a aplicação como se esta fosse um sistema desktop tradicional, o RIA baseia-se na premissa de rodar na máquina do usuário todo o processamento de interface gráfica, deixando para o lado do servidor o processamento da lógica de negócio.(MONTEIRO, 2008)

As aplicações que se utilizam do RIA “se assemelham as aplicações desktop, fazendo com que seu uso seja mais fácil, oferecendo uma interface mais rica aos usuários”. (LOOSLEY, 2006). Segundo BRIDEE, RIA é mais que uma tecnologia, é um conceito, reforçando que RIA é uma combinação de interatividade e funcionalidade do desktop com a flexibilidade e abrangência da Web.

Como ferramentas RIA pode-se citar: Adobe Flash, Flex e Air, Microsoft SilverLight, GoogleWebToolkit, JavaFX, OpenLazlo, e Frameworks baseados em Ajax.

3 TECNOLOGIAS RIAS

Neste capítulo teremos uma descrição das tecnologias estudadas como também a uma pequena comparação entre as mesmas, que mostra os motivos da escolha em utilizar as tecnologias GWT-EXT e Adobe Flex para o desenvolvimento da aplicação.

3.1 GOOGLE WEB TOOLKIT (GWT)

Desenvolvido dentro do Google e liberado pelos seus desenvolvedores como software livre o GWT pode ser escrito totalmente com código Java, utilizando-se de ambientes de desenvolvimento integrados de alta qualidade, tipagem estática e debug ativo, tendo um compilador especial que transforma o código Java em código JavaScript, para que ele possa executar no navegador cliente.

Vejamos alguns pontos importantes no uso desta ferramenta segundo Maurício Linhares de Aragão Junior (2007).

Criação do projeto: Executar o utilitário `projectCreator -ant GWTTutorial -eclipse GWTTutorial`, esse comando vai instruir a ferramenta a criar um “buildfile” do Ant com o nome `GWTTutorial.ant.xml` e os arquivos de configuração necessários para transformar esse projeto em um projeto do Eclipse. Depois de criados os arquivos, você pode importar esse projeto para dentro do Eclipse utilizando a importação de projetos.

Alguns arquivos criados importantes são: “`applicationCreator`”, que monta uma aplicação “esqueleto” ; “`GWTTutorial-shell`” que executa a aplicação no “hosted mode”, que é o “modo de desenvolvimento” das aplicações GWT, sendo que o GWT já tem um servidor

embutido (junto com um navegador) para teste de aplicações. Outro arquivo importante é o “GWTTutorial-compile”, que compila as classes Java e as transforma em JavaScript para que elas sejam executadas no navegador.

Padrão de design: Pacote criado deve, preferivelmente, terminar com o nome “client”, porque esse é o pacote onde vão estar as classes e interfaces que vão ser transformadas em JavaScript para executar no cliente. Além desse pacote, o GWT cria um pacote “public” onde vão estar disponíveis os arquivos de recursos para a aplicação (como os arquivos HTML, CSS e imagens). O outro pacote que pode existir é o “server” que guarda as classes que não são transformadas em JavaScript e são utilizadas através da API de invocação de serviços remotos do GWT. Os nomes das pastas são configuráveis, mas é uma boa prática manter os nomes padrão para evitar dores de cabeça futuras com configurações.

Invocação remota de serviços: A aplicação JavaScript que executa no cliente faz uma invocação a um serviço implementado no servidor (através de um *Servlet*) que pode então se utilizar de toda a expressividade e APIs do Java para retornar os dados para o cliente, como o JDBC, acesso a arquivos, EJB/Hibernate e outros. O GWT consegue enviar diretamente objetos comuns do Java, como todos os tipos primitivos, Strings e Dates (junto com suas representações como arrays), mas os tipos definidos pelo usuário precisam de um tratamento especial. As classes definidas pelo usuário que necessitem ser enviadas por invocações remotas devem implementar a interface de marcação `com.google.gwt.user.client.rpc.IsSerializable` e ter como propriedades apenas outros objetos que também implementem essa interface ou os objetos comuns do Java. Todos os objetos são inicializados na chamada do método “`onModuleLoad()`”, que cria os componentes e o objeto que vai fazer as chamadas ao serviço remoto.

Para criar um objeto que faz a chamada ao serviço, utiliza-se o método estático “`create()`” da classe “GWT”, passando como parâmetro o objeto *class* da interface real do serviço.

O objeto retornado é um objeto que implementa a interface assíncrona do serviço.

Depois de retornado o objeto, tem-se que fazer um “cast” dele para o tipo `com.google.gwt.user.client.rpc.ServiceDefTarget` e dizer qual é a URL na qual o serviço deve ser invocado. Essa URL é o mapeamento do *servlet* que implementou o serviço remoto e essa URL deve ser absoluta.

Para não ter que definir diretamente o caminho do contexto, utiliza-se o método estático `GWT.getModuleBaseURL()`, que retorna o caminho absoluto até a aplicação definida (com “/” no final) fazendo a concatenação com a URL do mapeamento do *servlet* na aplicação.

Douglas José Soares Rodrigues 2008 conclui que “o GWT mostra-se um framework extremamente eficiente e bem escrito, além de ser uma solução robusta para o desenvolvimento AJAX com o suporte de uma empresa renomada”.

3.2 EXT JS

Ext provê uma interface rica, fácil de usar, muito parecidas como as que você encontraria em uma aplicação desktop. Isto permite que desenvolvedores WEB concentrem na funcionalidade da aplicação web ao invés de suas ressalvas técnicas (FREDERICK; RAMSAY; BLADES).

Uma grande biblioteca JavaScript poderia ser um resumo do Ext, mas ele vai além disso, não só oferece um vasto leque de componentes como painéis, *grids*, *forms*, menus, janelas, abas, botões, etc., mas também dá outra perspectiva à aplicação com seus três pontos fortes:

- Ampla compatibilidade em navegadores: as problemáticas diferenças na interpretação de código JavaScript e CSS entre os navegadores que atormentam os desenvolvedores de

interfaces WEB são tratadas automaticamente pela biblioteca que dá suporte aos principais navegadores: Internet Explorer 6+, Firefox 1.5+ (PC, MAC), Safari 2+. Opera 9+ (PC, MAC).

- Interfaces baseadas em eventos: um evento poderia ser uma ação feita pelo usuário, como o clique de um botão ou gerado pela própria aplicação, como o carregamento da aplicação ou o retorno de uma requisição de dados. São estes eventos que invocam as classes implementadas pelo desenvolvedor.
- AJAX: A biblioteca utiliza AJAX para a comunicação com o servidor de maneira simples e transparente. Isso permite, por exemplo, carregar um *grid* enquanto um *form* é preenchido.

Common Name ▲	Light	Price	Available	Indoor?
Adder's-Tongue	Shade	\$9.58	Apr 13, 2006	<input checked="" type="checkbox"/>
Anemone	Mostly Shady	\$8.86	12/26/06	<input checked="" type="checkbox"/>
Bee Balm	Shade	\$4.59		
Bergamot	Shade	\$7.16		
Black-Eyed Susan	Sunny	\$9.80		
Bloodroot	Mostly Shady	\$2.44		
Blue Gentian	Sun or Shade	\$8.56		
Buttercup	Shade	\$2.57		
Butterfly Weed	Sunny	\$2.78		
California Poppy	Sunny	\$7.89		

Figura 1 – Um painel com um *grid* editável onde um calendário dinâmico ajuda o usuário a escolher a data para o campo.

Fonte: Site oficial Ext JS. Disponível em <<http://www.extjs.com/deploy/dev/examples/grid/edit-grid.html>>. Acesso em 06 de Outubro de 2009.

3.3 GWT-EXT

Desde o lançamento do Google Web Toolkit (GWT) e *frameworks* de *widgets* RIA relacionados como o Ext GWT, o desenvolvimento de RIA agora é muito mais fácil e flexível. Então, se você consegue construir aplicações Java, você pode construir rapidamente uma aplicação rica para internet de classe empresarial (SLENDER, GRANT).

Baseado na biblioteca JavaScript Ext JS o GWT-EXT é uma poderosa biblioteca de *widgets* como *grids*, painéis, *forms*, etc., para a plataforma GWT.

Basicamente o GWT-EXT é uma migração dos componentes da biblioteca EXT JS para a plataforma GWT, o que permite a utilização da linguagem de programação Java no lugar da linguagem JavaScript, que é utilizada no EXT JS.

Veja abaixo o código de um mesmo exemplo simples utilizando o Ext JS e depois utilizando o GWT-EXT:

```
Ext.get('mb8').on('click', function(){
    Ext.MessageBox.alert('Status', 'Changes saved successfully.', showResult);
});
```

Figura 2 – Utilizando Ext JS o código adiciona em um botão de id “mb8” a função que mostra uma caixa de alerta.

Fonte: Site oficial Ext JS. Disponível em <<http://www.extjs.com/deploy/dev/examples/message-box/msg-box.html>>. Acesso em 07 de outubro de 2009.

```
Button button7 = new Button("Show Me", new ButtonListenerAdapter() {
    public void onClick(Button button, EventObject e) {
        MessageBox.alert("Changes saved successfully");
    }
});
```

Figura 3 – Utilizando GWT-EXT o código cria um botão com o texto “Show Me” e adiciona nele a função que mostra uma caixa de alerta.

Fonte: Site oficial GWT-EXT. Disponível em <<http://www.gwt-ext.com/demo/#messageBox>>. Acesso em 07 de outubro de 2009.

Apesar de ter apresentado uma boa recepção pela comunidade de programadores WEB, o

projeto GWT-EXT não terá continuação. No fórum oficial da comunidade a equipe do GWT-EXT está aconselhando seus usuários a migrarem para outra biblioteca RIA-GWT como a SmartGWT. Isto se deve ao fato da licença do projeto Ext JS ter sido alterada, que até a versão 2.0.2 mantinha a licença LGPL.

3.4 LASZLO

Laszlo é um projeto de código aberto distribuído sob a proteção de uma OSI (*certified common public licence*) de autoria da IBM. Sua plataforma consiste na linguagem de programação LZX e no compilador OpenLaszlo Server. Este compilador é um *servlet* Java que compila o código LZX em DHTML ou SWF.

Conforme André Luís Monteiro 2008, “a *engine* de renderização do OpenLaszlo é uma aplicação Java para Web, portanto deve ser instalada em um container web como o Tomcat, ou num servidor de aplicação Java EE. Em tempo de execução o servidor Laszlo “compila” os arquivos fontes LZX para um aplicativo Flash (.SWF) que é enviado ao cliente”.

Ainda segundo Luís Monteiro 2008, “ao desenvolver aplicações OpenLaszlo, geralmente criamos primeiro toda a estrutura estática da aplicação, definindo os componentes visuais e layout, e depois adicionamos o comportamento e o processamento de evento”.

Para finalizar, Monteiro 2008, descreve OpenLaszlo como uma aplicação “funcional que vem ganhando apoio crescente da comunidade de desenvolvedores e integra-se especialmente bem com a tecnologia Java”

3.4.1 LZX

Linguagem orientada a objetos, permitindo definir classes e métodos e usar herança e polimorfismo. Utiliza-se de XML declarativa para definição de cada componente e de marcação do tipo HTML podendo receber parâmetros.

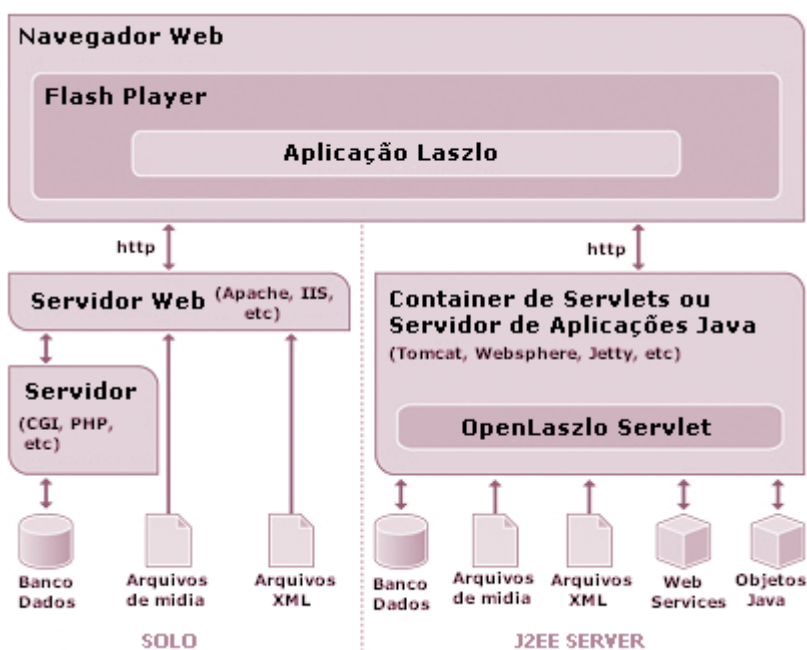


Figura 4 - Arquitetura da plataforma OpenLaszlo

Fonte: http://www.mundojava.com.br/NovoSite/images/fig01_laszlo_arquitetura.gif

3.5 ADOBE FLEX 2

Em 2004 a Macromedia Flex, hoje conhecida como Adobe Flex, divulgou o lançamento do produto Flex, hoje está com cinco anos de vida e já atinge um alto grau de utilização pelas empresas de sistemas Web.

Sua proposta de página única desbanca o estilo antigo de páginas em HTML com

links, onde a navegação consiste em página por página, com inúmeras requisições e possibilidades de erros como página não encontrada. Flex destaca-se pela a notável diferença de usabilidade entre as tecnologias usualmente empregadas para desenvolvimento de aplicações Web.

Em sua estrutura a Adobe escolheu integrar-se com a IDE Eclipse, visto que os desenvolvedores possuem afinidade com esta, facilitando assim sua adaptação. Para tal basta baixar o *plug-in* do Flex Builder, que requer licença.

O *runtime* do Flex é o Flash, requerendo a instalação de no mínimo o Flash 9 *player*. Apesar de rodar em Flash, o Flex não se caracteriza como um programa de animação.

Para desenvolvimento das aplicações a Adobe desenvolveu o Flex Builder, que provê um ambiente no qual o desenvolvedor e o *designer* trabalham juntos: o desenvolvedor com a linguagem ActionScript, e o designer com a linguagem MXML.

A linguagem MXML, derivada do XML, fornece suporte a estilos CSS (*Cascading Style Sheets*), facilidades em alternar cores e navegar entre diferentes funcionalidades. Com o Flex Builder o *designer* visualiza o código da aplicação e seu *design*, podendo redimensionar e editar o código MXML.

O adobe Flex é uma solução completa para criar e fornecer aplicações ricas, robustas, interativas e que possibilizem uma interface mais anigável e intuitiva para o usuário. Martinelli e Terracini.

Composto pelos seguintes itens:

Adobe Flex 2 SDK - Conjunto de classes (*framework*), compilador, *debugger*, duas linguagens de programação MXML e Actionscript além de incluir códigos fontes do conjunto de classes para customização pelo desenvolvedor.

Adobe Flex Builder 2 (IDE).

Adobe Flex Data Services 2 - Serviço de mensagem que roda em um servidor JEE compatível ou em *contêiner Servlet*.

Adobe Flex Charting 2 - Funcionalidade que facilita a visualização de um conjunto de dados no formato de gráficos de duas dimensões.



Figura 5 - Produtos da linha Flex 2

Fonte: http://www.mundojava.com.br/NovoSite/images/f4_20_capa.jpg

O modelo de aplicativo pode ser representado pelo modo como ele é executado no servidor. Sendo estes: Client-side Only Applications (executada no cliente não utilizando recursos do servidor), Data Access With HTTPServices and WebServices (comunicação com o servidor através de chamadas HTTP ou utilizando WEB *services*) e Data Access With Flex Data Services (vantagens no acesso ao servidor, tais como, sincronização, segurança e comunicação). Segundo Martinelli e Terracini “a forma mais simples é ler em XMLs via conexão HTTP, porém também é possível utilizar o padrão SOAP para conectar a Web Services.”

3.5.1 MXML e ActionScript

A linguagem de programação MXML foi desenvolvida pela Macromedia para a programação de interfaces para seu novo servidor de aplicações Flex, que funciona sob servidores J2EE como JRun, Websphere, WebLogic ou Tomcat. O MXML inclui um conjunto de *tags* específicas como DataGrid, Tree, TabNavigator, Accordion, Menu e também permite a criação de novos componentes, que são renderizados pelo Flash Player. Já a programação de respostas a eventos do usuário é feita na linguagem ActionScript, baseada no padrão ECMA-262. Sua codificação é similar ao HTML pois também utiliza *tags*, mas sua grande vantagem é a possibilidade da utilização de Webservices, permitindo assim acesso a dados.

3.6 MICROSOFT SILVERLIGHT

Projeto desenvolvido pela Microsoft que suporta AJAX, Python, Ruby e as linguagens .NET, Visual Basic e C#. É composto por um *runtime*, um SDK e ferramentas de desenvolvimento e design. A linguagem utilizada para o design é XAML (*Extensible Application Markup Language*), sendo que o *runtime* executa esta linguagem e utiliza o JavaScript para manipulação dos objetos Silverlight.

O objetivo do XAML é permitir que o designer e o desenvolvedor possam trabalhar utilizando a mesma tecnologia.

Com Silverlight, o designer pode utilizar ferramentas que expressam um design em XAML, e o desenvolvedor pode pegar este XAML, ativá-lo com código, e rodar em um programa (MORONEY, LAURENCE).

A camada de apresentação do Silverlight é feita em HTML, que contém chamadas para instanciar o plug-in Silverlight. Enquanto usuários interagem com a página, ela gera eventos

que são capturados pelas funções escritas em JavaScript. Em troca, o código escrito em JavaScript pode realizar chamadas de métodos para os elementos dentro do conteúdo Silverlight para manipulá-lo, adicionar novo conteúdo, ou remover conteúdo existente. Finalmente, XAML pode ser lido pelo plug-in e renderizado. O próprio XAML pode existir escrito dentro da página, externamente num arquivo estático, ou XAML dinâmico retornado por um servidor (MORONEY, LAURENCE).

Uma aplicação desenvolvida em Silverlight simples é composta por um arquivo XAML, um arquivo JavaScript e uma página HTML. A seguir podemos ver um pequeno exemplo de uma aplicação desenvolvida em SilverLight.

3.6.1 XAML

Abaixo podemos ver um exemplo de uma interface XAML simples, que cria um *canvas* com uma caixa de texto escrito “Hello World!”. Todo arquivo XAML tem a *tag* “*canvas*” como raiz do arquivo XML, no caso abaixo vemos que esta *tag* também tem algumas propriedades declaradas como largura, altura, cor de fundo e nome da página. Dentro da *tag* “*canvas*”, temos uma *tag* da caixa de texto com as propriedades largura, altura, posição relativa à esquerda do *canvas*, posição relativa ao topo do *canvas*, texto, quebrar linha e nome da caixa de texto.

```
<Canvas
  xmlns="http://schemas.microsoft.com/client/2007"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Width="640"      Height="480"
  Background="White"
  x:Name="Page">
  <TextBox Width="195" Height="42" Canvas.Left="28"
Canvas.Top="35"
      Text="Hello World!" TextWrapping="Wrap"
  x:Name="txt" />
</Canvas>
```

3.6.2 JavaScript

O conteúdo em JavaScript consiste na implementação de uma função que carrega a interface XAML e das funções que lidam com a lógica da aplicação. No exemplo abaixo a regra de negócio simplesmente altera o texto da caixa de texto para “You clicked me!”.

```

/**
 * Função que carrega a interface XAML
 */
Function createSilverlight() {
    Silverlight.createObjectEx({
        source: "Page.xaml",
        parentElement:
document.getElementById("SilverlightControlHost"),
        id: "SilverlightControl",
        properties: {
            width: "100%",
            height: "100%",
            version: "1.0"
        },
        events: {
            onLoad: handleLoad
        }
    });
}

/**
 * Regras de negócio
 */
var SilverlightControl;
var theTextBlock;
function handleLoad(control, userContext, rootElement) {
    SilverlightContro = control;
    theTextBlock =
SilverlightControl.content.findName("txt");
    theTextBlock.addEventListener("MouserLeftButtonDown",
"txtClicked");
}
Function txtCliked(sender, args) {
    theTextBlock.Text = "You clicked me!";
}

```

3.6.3 HTML

O conteúdo em HTML da aplicação é encarregado de chamar a função JavaScript que inicia a interface. Segue um exemplo de um documento:

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3c.org/TR/1999/REC-html401-19991224/loose.dtd">
<HTML xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Example</title>
  <script type="text/javascript" src="Silverlight.js">
  </script>
  <script type="text/javascript"
    src="CreateSilverlight.js"></script>
  <script type="text/javascript" src="code.js">
  </script>
  <style type="text/css">
    .silverlightHost {
      height: 480px;
      width: 640px;
    }
  </style>
</head>
<body>
  <div id="SilverlightControlHost" class="silverlightHost">
    <script type="text/javascript">
      createSilverlight();
    </script>
  </div>
</body>
</html>

```

3.7 JAVAFX

Nova família de produtos Java projetada para a criação de conteúdo multimídia para dispositivos como PDA's, entre outros. Inicialmente a plataforma consiste em dois produtos: JavaFX Mobile e JavaFX Script, uma nova linguagem de script que permite criar interfaces de forma declarativa.

O principal objetivo da Sun para esta plataforma é replicar o sucesso alcançado pelo

Java no meio corporativo, nas soluções voltadas para os consumidores finais.

JavaFX Mobile é um sistema operacional para celulares, smartphones e PDA's. Este novo OS é baseado em um kernel Linux com implementações do Java SE e Java ME. Segundo a Sun, o JavaFX Mobile usará um modelo OEM que permitirá o *rebrand* do sistema pelas operadoras de telefonia, fabricantes e qualquer outra empresa que precise de um OS.

Focada no desenvolvimento para o consumidor final e RIA, o JavaFX Script é uma linguagem de script baseada nos componentes do *Swing*. Seus principais concorrentes para RIA são: Ajax, Adobe Flex e Microsoft Silverlight, porém a Sun ainda não deixou claro se desenvolvimento RIA com JavaFX será feito utilizando Applets, Web Start ou até mesmo um novo produto específico para RIA.

3.7.1 JavaFX Mobile

O novo OS da Sun será lançado sob licença GPL e comercializado sob o modelo OEM. Além dos serviços básicos de mensagens, um browser com suporte a música e vídeo, o JavaFX Mobile vem acompanhado da API Java, ferramentas para GUI, software update, entre outros.

Na figura abaixo, serviços e aplicações oferecidos no JavaFX Mobile:



Figura 6 - JavaFX Mobile nos PDAs
Fonte: Augusto Uehara, 2007

3.7.2 JavaFX Script

Sendo uma linguagem que permite o desenvolvimento de aplicações RIA, desktop e dispositivos móveis, sua maior vantagem é o fato de ser uma linguagem declarativa, estática que aproveita a mesma estrutura de código, reuso e encapsulamento do Java e permite chamadas diretas aos métodos da API Java disponíveis.

James Gosling definiu o JavaFX como “Orientada a interfaces altamente animadas”.

As promessas do JavaFX são:

- Eliminar alguns problemas de segurança e compatibilidade comuns em seus concorrentes;
- Tornar acessível o desenvolvimento RIA com Java aos designers e demais autores do conteúdo;
- Verdadeiramente multiplataforma;
- Portabilidade que apenas o Java oferece, permitindo interfaces consistentes desde Desktop até os dispositivos móveis.

Alguns pontos fracos são:

- O tamanho e complexidade da instalação. O Flash Player e o Silverlight Player possuem em torno de 5mb em suas maiores versões, enquanto a JRE possui 13mb em sua menor versão;

- Designers e autores de conteúdo já estão habituados com o Flash;

- O AJAX já possui seu lugar no mercado. Dificilmente o JavaFX tomaria o lugar do AJAX no Gmail ou no Yahoo e outros sites WEB 2.0 para consumidor final.

O maior problema do JavaFX para Java SE conforme Doederlein (2008) “é herdar as dificuldades já grandes do Java para aplicações Internet ricas: a instalação do JRE, a impopularidade das *applets*, o tempo de inicialização, e o consumo de memória”. Como pontos fortes o autor define a sintaxe especializada, novos *codecs* de mídia e *frameworks* para gráficos e animações, que permitem criar aplicações com pouco código e ótimo desempenho, além de possuir disponibilidade tanto para PCs quanto para dispositivos moveis.

Com o JavaFX é possível fazer chamadas para métodos e criar objetos de forma procedural, semelhante ao Java:

```
var janela = new Frame();
var botao = new JButton("clique aqui");
janela.getContentPane().add(botao);
botao.addActionListener(new ActionListener() {
    operation actionPerformed(evento){
        System.out.println("Você clicou o botão!");
    }
});
janela.pack();
janela.setVisible(true);
```

Porém a melhor forma de fazer uso do JavaFX é trabalhando com ele de forma declarativa, o mesmo código acima pode ser feito da seguinte maneira:

```
Frame {
    content : Button{
        text : "Clique aqui"
```



```

        action : operation() {
            System.out.println("Você clicou o botão!");
        }
    }
    visible : true
}

```

Outras características que chamam a atenção no JavaFX Script:

- Lista de compreensão e Arrays: a lista de compreensão fornece ao programador uma nova maneira de trabalhar com Arrays muito poderosa, similar ao SQL, permite funções como o filtro de números menores que 10 em uma lista de números. Ex.: “var menoresQueDez = numeros[. < 10]”.
- Concorrência: uma simples maneira de trabalhar e criar concorrências em seus programas com a utilização de palavras reservadas como “do” e “do later”.
- Classes e objetos: as classes também são criadas de forma declarativa. Diferente das classes em Java, em que os métodos são declarados dentro do corpo da classe, em JavaFX as funções e operações são declaradas dentro do corpo da classe, mas implementadas fora do corpo da classe.
- *Triggers*: Criadas para compensar algumas limitações do JavaFX, possuem quatro formas: criação de um objeto, inserir, apagar e atualizar um atributo.
- Reflexão: Como no Java, esta característica poderosíssima também está disponível no JavaFX com alguns recursos extras para facilitar seu uso.

3.7.3 Arquitetura JavaFX

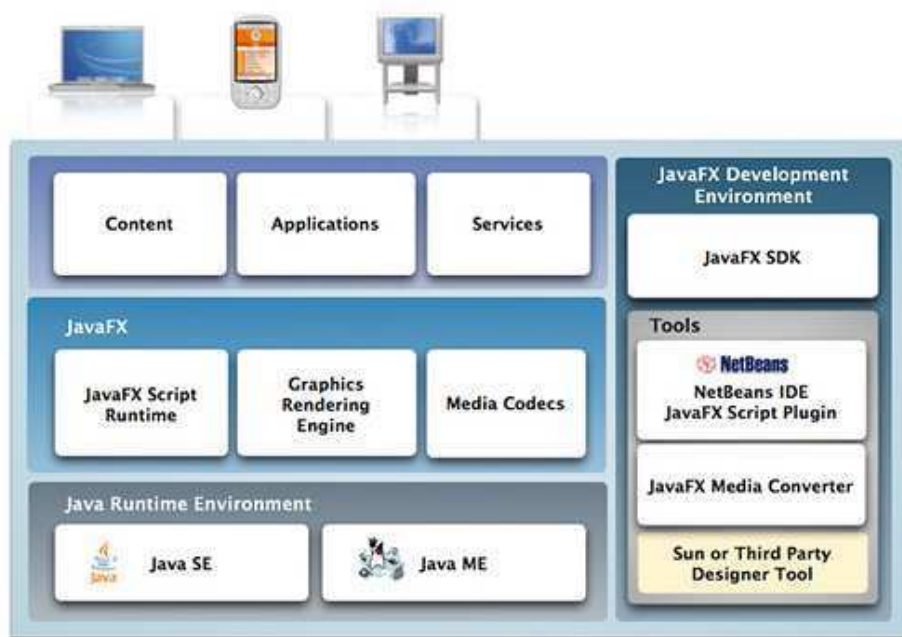


Figura 7 - Arquitetura JavaFX

Fonte: [HTTP://blogs.sun.com/chhandomay/resource/javaty_plataform.jpg](http://blogs.sun.com/chhandomay/resource/javaty_plataform.jpg)

3.8 COMPARAÇÃO

Tecnologia Características	Adobe Flex	GWT Ext	Java FX	Microsoft Silverlight	Laszlo
Suporte a Linguagem dinâmica	Sim	Sim	Não	Sim (JavaScript e IronPython)	Sim
Suporte a Linguagem estática	Sim	Sim	Sim	Sim	Sim
IDE de apoio	Eclipse(plugin: Flex Builder), Flex application.	Eclipse (plugins: WTP WEB TOOLS PROJECT) e googlipse IntelliJ VistaFei	Eclipse, NetBeans.	Visual Studio.	Eclipse, NetBeans.
Arquitetura	Ferramentas, estruturas, servidores e serviços.	Servidor Java, classes de interface compiladas em HTML, JavaScript e CSS.	Java SE e Java ME	Um RunTime, um SDK e ferramentas de desenvolvimento e design (Servidor, XAML, suporte a AJAX)	Servidor, compilador e runtimes (Em tempo e execução, o servidor Openlaszlo "compila" os arquivos fonte LZX para um aplicativo Web.)
Compatibilidade	Multiplataforma (linux, mac, windows)	Multiplataforma (solaris, linux, mac)	Multiplataforma (solaris, linux, mac, windows, windows mobile)	Windows, Mac OSX, Linux	Independente de sistema operacional
Servidor	J2EE com Jrun, Wepsphere, WebLogic ou TomCat.	Servidor WEB Java ou servidor com suporte Xmlhttp	Independente, necessita JVM	ASP .NET	Tomcat ou servidor de aplicação Java EE

Tecnologia	Adobe Flex	GWT Ext	Java FX	Microsoft Silverlight	Laszlo
Características					
Open Source	Não	Sim sob a licença Apache 2.0	Não, possui algumas dependências de componentes licenciados	Não	Sim. Distribuído sob a proteção de uma OSI (<i>Certified common public licence</i> de autoria da IBM)
Plataforma	Flash	Navegador WEB	Java SE e Java ME	Silverlight	Todos os browsers e plataformas
Posição no mercado	Dominante	Ascensão	Inicial	Inicial	Ascensão
Designer & desenvolvedores	Designer: MXML / Desenvolvedor : ActionScrit.	Designer: CSS / Desenvolvedor: Java	Fácil entendimento por ambos. Porém designers e autores estão mais habituados com o flash.	O design pode utilizar ferramentas que expressam um design em XAML, e o desenvolvedor pode pegar este XAML, ativá-lo com código, e rodar em um programa.	Funcional. Vem ganhando Apoio crescente da comunidade de desenvolvedores
RunTime	FLASH	HTML + JavaScript + CSS	JDK	FLASH	FLASH
Suporte a mídia	Sim	Não	codecs (JavaFX Movie - subset FLV, MP3)	Sim	Sim

Tecnologia	Adobe Flex	GWT Ext	Java FX	Microsoft Silverlight	Laszlo
Características	XMLs via conexão HTML ou SOAP	Sim, Java api	API - REST	REST – <i>Representation State Transfer</i> ou representação de transferência de Estado (Técnica de arquitetura de software para sistemas distribuídos)	Requisições HTTP para XML, SOAP, XML-RPC, e JavaRPC Services
Web Service					
Desenvolvido por	Adobe Systems Incorporated	Google	Sun Microsystems	Microsoft	IBM
Ano de criação	2004	2006	04/12/2008	2006	2003
Versão atual	3.4	1.7	JavaFx 1.5 e um pré release JavaFX 2.0	3	4.6

Tabela 1 - Comparação entre tecnologia RIA, estudo comparativo.

Fonte: Primária.

A tabela acima (tabela 1) traz um comparativo com cinco das grandes tecnologias RIAs disponíveis no mercado na atualidade. Para comparação foi elencado uma série de características das tecnologias, conforme segue:

Suporte a Linguagem dinâmica: Linguagem executada em tempo de execução.

Suporte a Linguagem estática: O compilador deve conhecer o tipo de uma variável ou método antes da execução do programa.

IDE de apoio: Ambiente de desenvolvimento integrado utilizado pela tecnologia.

Arquitetura: Tecnologias utilizadas, utilização de servidores, compilação e runtimes.

Compatibilidade: Integração com outras tecnologias.

Servidor: Servidores que a tecnologia comporta e utiliza.

Open Source: Tecnologia aberta para utilização, restrita.

Plataforma: Plataformas que suportam a tecnologia.

Posição no mercado: Frequência da utilização no mercado da tecnologia.

Designer & desenvolvedores: Como a tecnologia separa os desenvolvedores dos designers.

Runtime: Como funciona a leitura do código em tempo de execução.

Suporte a mídia: Tecnologia possui suporte a mídias. Exemplo: mp3.

Web Service: Tecnologia suporta integração com WebServices, se sim, como?

Desenvolvido por: Empresa desenvolvedora da tecnologia.

Ano de criação: Lançamento da tecnologia.

Versão atual: Última versão liberada do produto.

Desempenho: Velocidade da tecnologia, referente às outras.

A tabela 1 demonstra algumas das principais diferenças e características entre as tecnologias RIA's.

Para desenvolvimento deste trabalho poderíamos utilizar qualquer uma das cinco tecnologias acima descritas, porém optamos por utilizar duas: o Adobe Flex e o GWT-EXT. O Adobe Flex foi escolhido por ser uma tecnologia que domina o mercado e por ter como único requisito de execução o Flash Player, presente em 99% dos computadores com acesso à internet.

Já o GWT-EXT foi escolhido por utilizar Java e por ser uma tecnologia em ascensão. Outro ponto importante da escolha das tecnologias foi sua adequação com o Java e com o

ambiente de desenvolvimento (IDE) Eclipse, por este ser um ambiente muito utilizado por programadores Java.

4 APLICAÇÃO

Este capítulo aborda o projeto da aplicação e o desenvolvimento da mesma. No projeto da aplicação teremos uma descrição de como deverá ser a aplicação, enquanto o capítulo de desenvolvimento relata a implementação do projeto.

4.1 PROJETO DA APLICAÇÃO

Na etapa do projeto da aplicação iremos detalhar como deve ser a aplicação a ser implementada, independente da tecnologia utilizada. Esperamos com isto, comparar as duas tecnologias em pontos singulares de desenvolvimento.

O tipo de aplicação escolhido foi um “Dashboard” para a visualização de gráficos e relatórios de acesso a sites de internet, inspirado no Google Analytics. Este “Dashboard” ou painel será preenchido com vários quadrados que formam áreas para diferentes funcionalidades que mostram estatísticas.

As funcionalidades escolhidas a serem implementadas estão listadas abaixo, junto com algumas imagens que mostram exemplos para a aplicação.

- Gráfico em linha de visitas;



Figura 8 – Exemplo de gráfico em linhas de visitas.

Fonte: Google Analytics.

- Gráfico em pizza das fontes de tráfego;

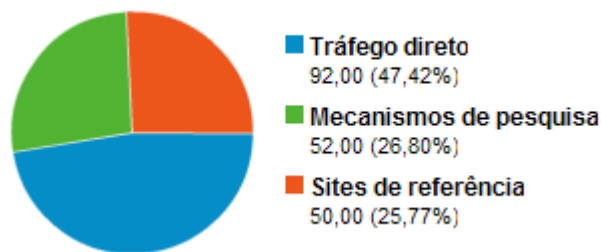


Figura 9 – Exemplo de gráfico em pizza das fontes de tráfego.

Fonte: Google Analytics.

- Gráfico em barras dos navegadores utilizados no site;



Figura 10 – Exemplo de gráfico em barras dos navegadores utilizados.

Fonte: Google Analytics.

O sistema terá um fluxo bem simples, que resultará em apenas dois níveis de navegação: o “Dashboard” no nível 1 e cada funcionalidade no nível 2. O “Dashboard” mostrará todas as funcionalidades, sendo que cada uma delas mostrará uma visão geral da mesma. Clicando na funcionalidade estaremos acessando o nível 2, que mostra a funcionalidade por completa.

Todo o sistema tentará utilizar os efeitos padrões de cada uma das tecnologias utilizadas.

4.1.1 Diagrama das Interfaces da Aplicação

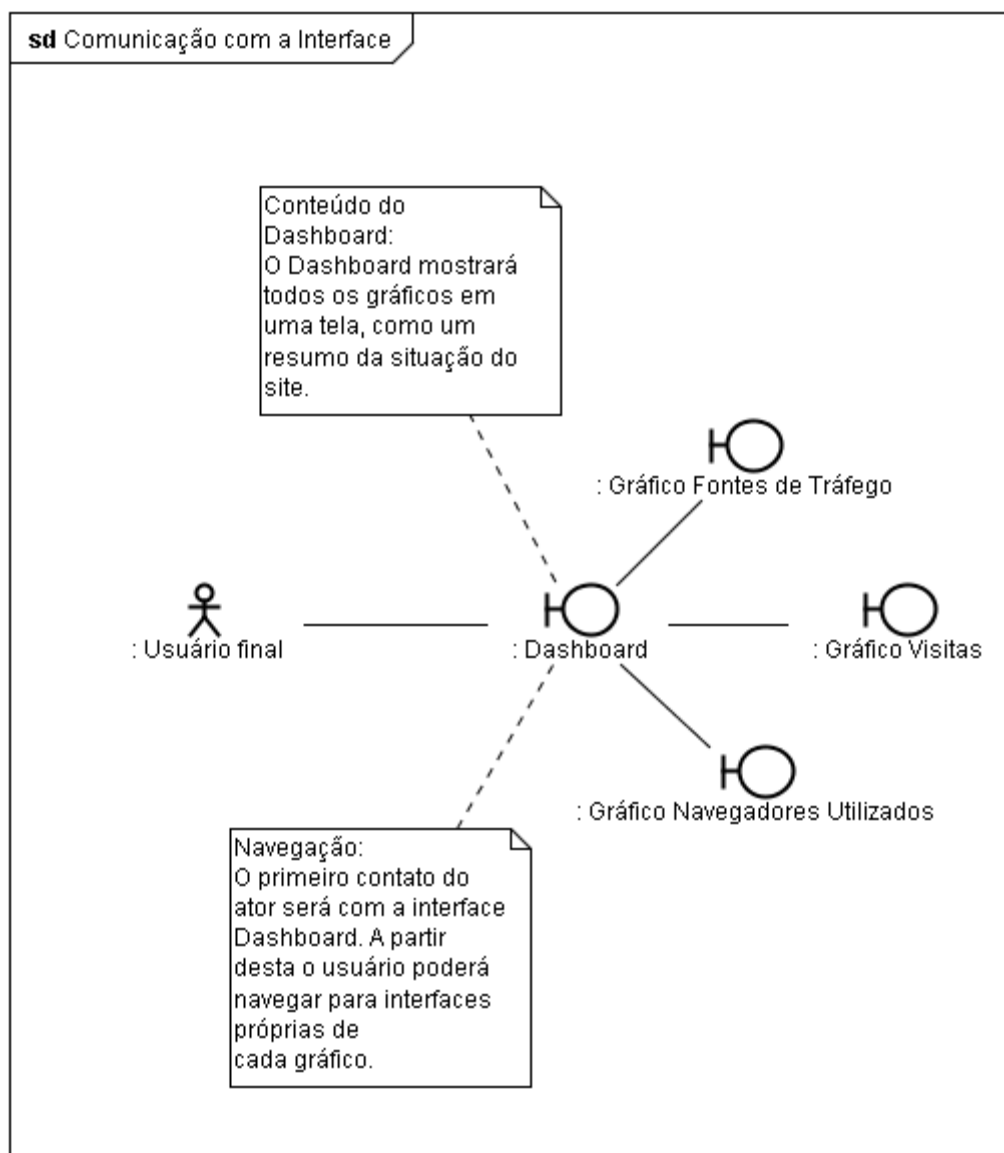


Figura 11 – Diagrama da interface da aplicação.

4.2 DESENVOLVIMENTO DA APLICAÇÃO

Nesta etapa iremos descrever o desenvolvimento da aplicação, que é formada pela integração das tecnologias Adobe Flex e GWT-EXT em uma mesma aplicação, pois compartilham pacotes e classes no servidor apesar de não compartilharem classes nos pacotes utilizados no lado cliente.

A aplicação roda dentro do servidor JBoss, que foi escolhido por ser um servidor de código aberto e também muito robusto, que suporta a tecnologia EJB e garantiria qualquer requisito para a integração das tecnologias Flex e GWT-EXT.

O desenvolvimento da aplicação foi feito utilizando a IDE de apoio Eclipse, para a implementação geral e controle de versões, e Flex Builder, utilizada na construção da aplicação em Flex.

4.2.1 Plataforma da Aplicação

A aplicação é executada sobre o servidor de aplicação JBoss, ao ser acessada, o usuário faz um contato com o contexto WEB, o pacote “cliente” da aplicação, que executa o código da tecnologia acessada. Este código se comunica com o servidor via requisições HTTP aos *servlets*, localizados nos módulos GWT ou Flex.

No caso do GWT, estes *servlets* ficam transparentes ao desenvolvedor, dando a

impressão que o código GWT do contexto WEB se comunica diretamente com o módulo GWT no servidor.

Os módulos GWT e Flex se comunicam com os EJB's da aplicação, localizados no módulo EJB e gerenciados pelo servidor de aplicação.

Os EJB's então acionam o pacote de utilitários criados para facilitar o acesso aos dados, que são armazenados no formato XML e acessados utilizando a biblioteca *XPath* incorporada à biblioteca padrão do Java em sua versão 1.6.

A plataforma da aplicação e seu fluxo são representados na figura abaixo:

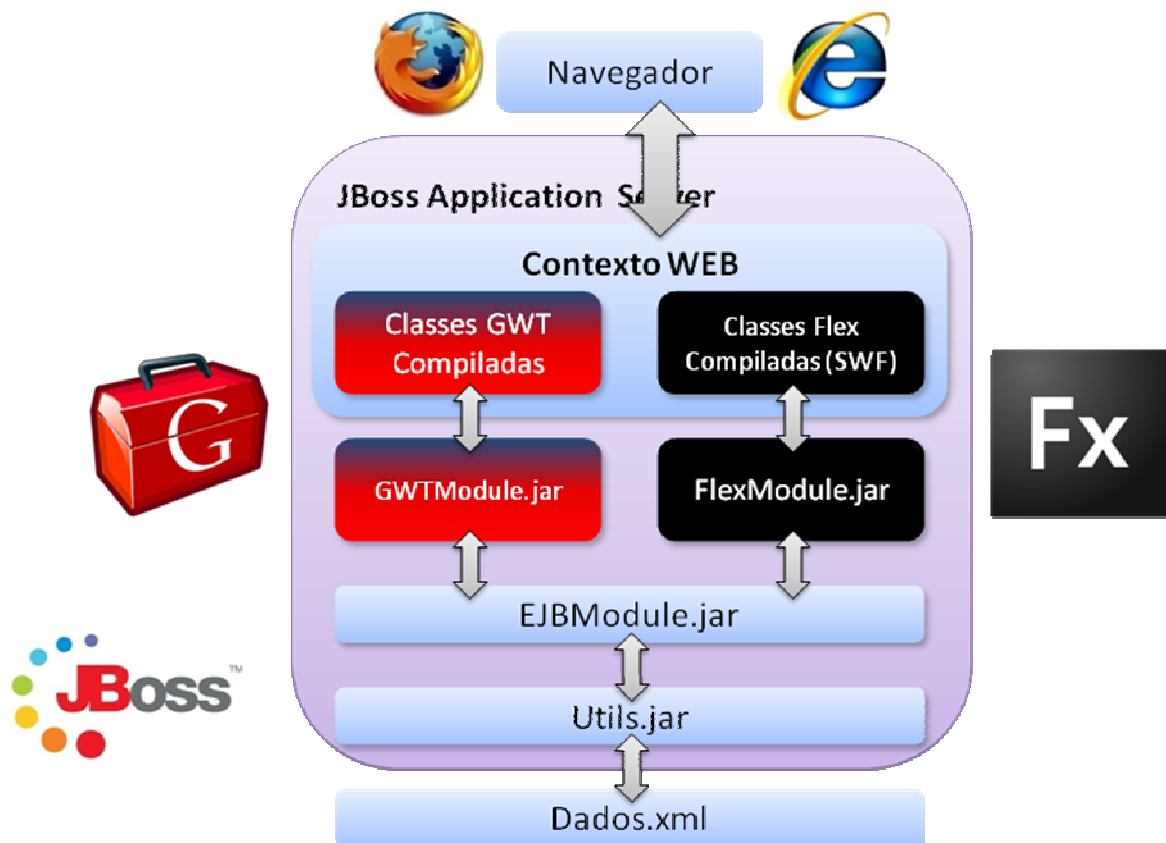


Figura 12 – Plataforma da aplicação

4.2.2 Configuração da Aplicação

Criado o projeto Java no Eclipse iniciamos a configuração da aplicação no servidor, que inclui estruturar a mesma no padrão J2EE, além de criar os arquivos de configuração da aplicação no JBoss. A figura 13 representa a estrutura da aplicação:

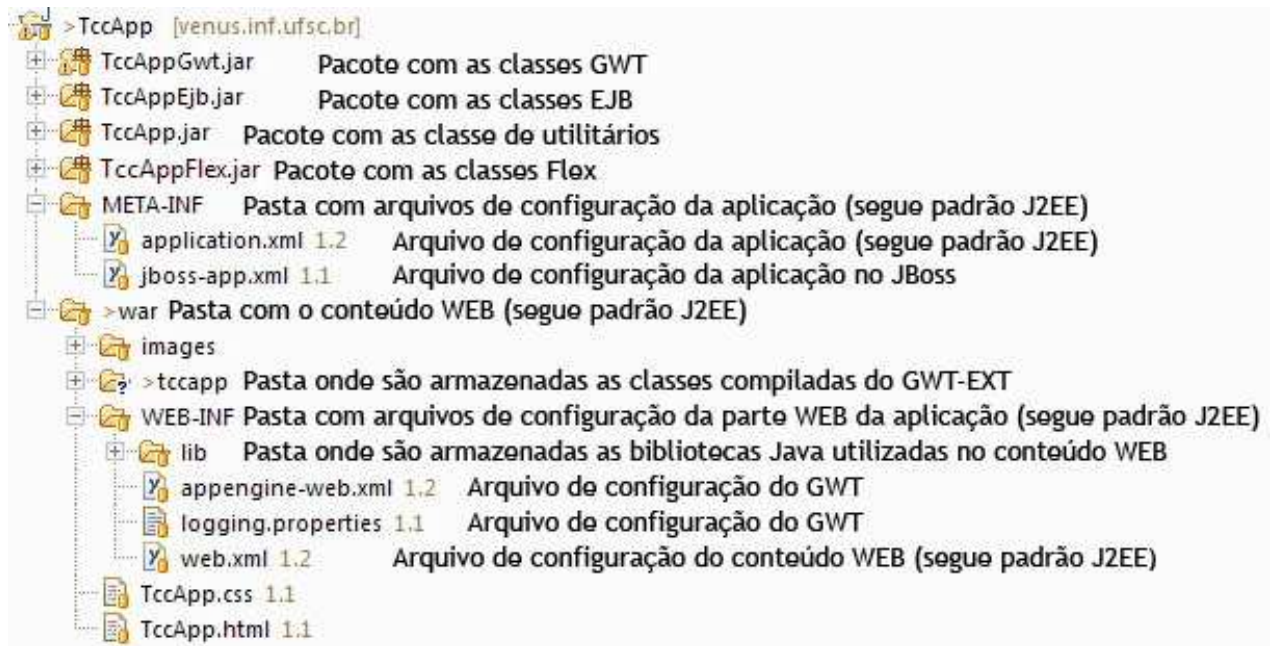


Figura 13 – Estrutura da aplicação, baseada no padrão J2EE

4.2.2.1 Arquivo de configuração: *application.xml*

Neste arquivo são feitas configurações dos módulos da aplicação, permitindo que um módulo possa ser acessado por classes localizadas em outro módulo. Também define como cada um deve ser carregado. No exemplo apresentado na Figura 14 podemos ver a configuração do contexto WEB, do pacote de classes utilitárias e do pacote EJB.

```

<module>
  <web>
    <web-uri>war</web-uri>
    <context-root>/tccapp</context-root>
  </web>
</module>
<module>
  <java>TccApp.jar</java>
</module>
<module>
  <ejb>TccAppEjb.jar</ejb>
</module>
</application>

```

Figura 14 - Configuração dos módulos no application.xml

4.2.2.2 Arquivo de configuração: web.xml

Seguindo o padrão J2EE, neste arquivo encontramos as configurações do contexto WEB. Abaixo podemos ver algumas configurações de *servlets* e do arquivo de boas vindas da aplicação.

```

<servlet>
  <servlet-name>testeServiceServlet</servlet-name>
  <servlet-class>br.ufsc.inf.tccapp.gwt.server.services.TesteServiceImpl</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>testeServiceServlet</servlet-name>
  <url-pattern>/tccapp/testeservice</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>visitorsChartDataServiceServlet</servlet-name>
  <servlet-class>br.ufsc.inf.tccapp.gwt.server.services.VisitorsChartDataServiceImpl</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>visitorsChartDataServiceServlet</servlet-name>
  <url-pattern>/tccapp/visitorschartdataservice</url-pattern>
</servlet-mapping>

<!-- Default page to serve -->
<welcome-file-list>
  <welcome-file>TccApp.html</welcome-file>
</welcome-file-list>
</web-app>

```

Figura 15 – web.xml: configuração do contexto WEB

4.2.3 Desenvolvimento GWT-EXT

Esta etapa retrata o desenvolvimento do módulo GWT da aplicação, que utiliza a biblioteca GWT-EXT rodando em uma plataforma GWT. Esta plataforma trabalha ao todo com quatro tecnologias:

- Java: utilizada na maior parte do código, na construção da camada de negócios, da camada de acesso aos dados e também da camada de interface. Esta última, porém, também utiliza outras linguagens.
- HTML: é o *container* da camada de interface. No HTML é inserido o código JavaScript resultante da compilação do módulo GWT que se comunica com a camada de negócios. Também invoca a utilização da tecnologia CSS.
- JavaScript: em geral é gerado no processo de compilação do módulo GWT, entretanto ainda permite que o desenvolvedor crie e utilize suas próprias funções, se houver necessidade. No caso da aplicação, não foi implementada nenhuma função JavaScript.

Este também é o módulo responsável pela comunicação com o servidor, via requisições XML (AJAX), invoca métodos implementados em Java, localizados no módulo GWT, na camada de negócios.

- CSS: aplica estilos no *container* HTML, cores, posicionamento, tamanhos. Não altera o que o usuário final vê na interface, mas sim como ele a vê.

O desenvolvimento teve início com a criação do projeto base, que auxilia na configuração do módulo GWT. É criado a partir do Google Plug-in for Eclipse, que inicia um passo-a-passo de configuração. Nele é informado que o projeto utilizará o Google Web Toolkit e também qual será o pacote inicial do módulo GWT, que no nosso projeto é o pacote

br.ufsc.inf.tccapp.gwt.

Com estas informações, o passo-a-passo cria o projeto e seus arquivos necessários para ser compilado e executado com sucesso:

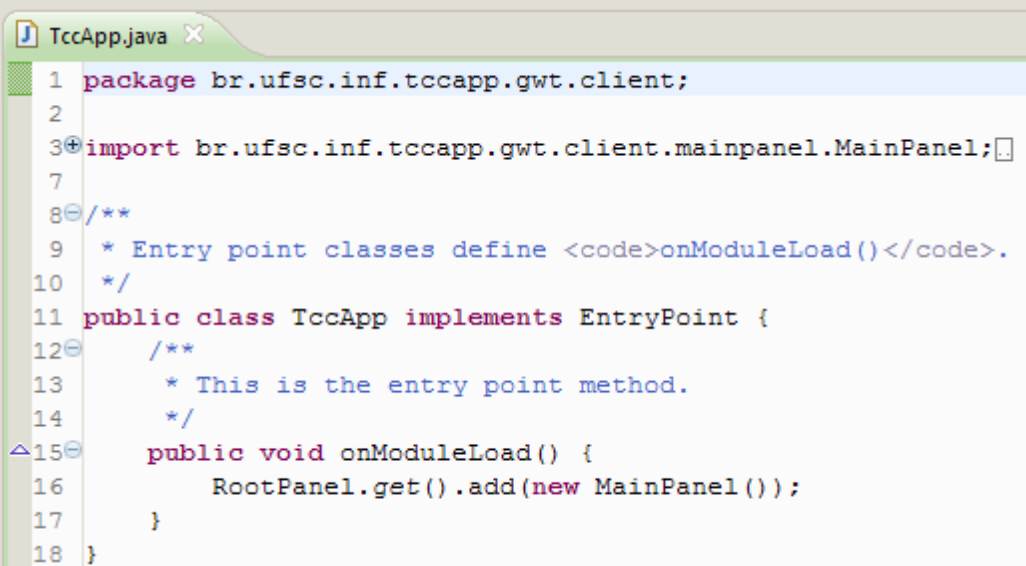
1. TccApp.gwt.xml;
2. TccApp.java;
3. jdoconfig.xml;
4. log4j.properties;
5. appengine-web.xml;
6. logging.properties;
7. web.xml;
8. TccApp.css;
9. TccApp.html;
10. Pasta lib com bibliotecas para o contexto web;
11. Pasta tccapp com os arquivos GWT compilados;

Dos arquivos acima, apenas os quatro primeiros estão dentro do módulo GWT, os outros fazem parte do contexto WEB. Alguns deles não são relevantes, como os arquivos de configuração de *logging*, utilizados apenas para que o projeto possa informar possíveis erros na compilação do módulo GWT. Outros são relevantes, como o arquivo TccApp.gwt.xml que é responsável pelas principais configurações do módulo GWT, declara as bibliotecas que devem ser utilizadas na compilação e execução do projeto como a própria biblioteca base do GWT e também permite a adição de bibliotecas extras. Também define qual é a classe ponto de entrada do projeto, a qual deve retornar o objeto que será a base da interface da aplicação.

Outras configurações definidas neste arquivo são: qual o tema de estilo será utilizado na interface e importações que devem ser feitas na página WEB base da aplicação, como estilos

CSS ou *scripts* JavaScript.

A classe ponto de entrada, TccApp.java, após declarada no TccApp.gwt.xml é a classe que é invocada quando a aplicação é carregada. Nisso, o desenvolvedor terá a possibilidade de acionar painéis e outros elementos de interface, como pode ser visto abaixo:



```

1 package br.ufsc.inf.tccapp.gwt.client;
2
3 import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;
4
5
6
7
8 /**
9  * Entry point classes define <code>onModuleLoad()</code>.
10 */
11 public class TccApp implements EntryPoint {
12     /**
13      * This is the entry point method.
14      */
15     public void onModuleLoad() {
16         RootPanel.get().add(new MainPanel());
17     }
18 }

```

Figura 16 – A classe ponto de entrada TccApp.java

Já no contexto WEB, encontramos arquivos importantes, como o arquivo web.xml que define os *servlets* da aplicação. No caso do módulo GWT, estes *servlets* representam os serviços implementados pelo desenvolvedor. Além dos *servlets*, neste arquivo também é definida a página que inicia a aplicação.

Para o módulo GWT, a página de início é a TccApp.html, que quando carregada, invoca a classe ponto de entrada.

No cabeçalho desta página são declarados, além de título e tipo do conteúdo, alguns arquivos importantes como o arquivo de estilo do GWT, TccApp.css e o script base do módulo, tccapp.nocache.js, que é criado toda vez que o projeto é compilado.

Dentro do elemento *body* da página está inserido apenas um elemento opcional que

permite que o usuário utilize as funções voltar e avançar do navegador na aplicação. Entretanto é possível declarar qualquer elemento HTML normalmente, que podem ser tratados pelo módulo GWT por intermédio de seus respectivos id's, que identificam cada elemento.

Para utilizarmos também a biblioteca GWT-EXT e a biblioteca GWT-EXT para gráficos foram necessárias algumas pequenas configurações.

O primeiro passo foi configurar o arquivo TccApp.gwt.xml para importar a biblioteca GwtExt e também a biblioteca de gráficos. Neste também importamos alguns *scripts* e arquivos de estilos do GWT-EXT. Estas configurações podem ser vistas na imagem abaixo.



```

17 <!-- Other module inherits -->
18 <!-- Inherit the core GWT-Ext Toolkit stuff. -->
19 <!-- GWT-EXT - Importa a biblioteca do GWT-Ext Toolkit. -->
20 <inherits name='com.gwttext.GwtExt' />
21 <inherits name='com.gwttext.Charts' />
22
23 <!-- Specify the app entry point class. -->
24 <entry-point class='br.ufsc.inf.tccapp.gwt.client.TccApp' />
25
26 <!-- GWT-EXT - Arquivos de estilo e scripts da biblioteca EXT -->
27 <stylesheet src="js/ext/resources/css/ext-all.css" />
28 <script src="js/ext/adapter/ext/ext-base.js" />
29 <script src="js/ext/ext-all.js" />
30</module>

```

Figura 17 – TccApp.gwt.xml – Configurações para a utilização da biblioteca GWT-EXT.

Depois foi necessário adicionar a biblioteca GWT-EXT Java nas referências do projeto, copiar a biblioteca GWT-EXT JavaScript em uma pasta criada dentro do módulo GWT, além de copiar o arquivo do Adobe Flash necessário para utilizar os gráficos, como mostra a figura abaixo:

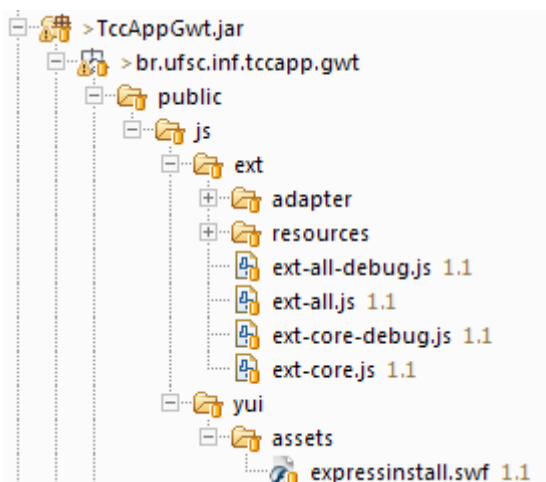


Figura 18 – Arquivos JavaScript da biblioteca GWT-EXT.

Esses arquivos JavaScript do GWT-EXT são colocados dentro desta pasta *public* para serem compilados junto com o módulo GWT, caso contrário não serão exportados para o contexto WEB, o que causaria um erro na execução do programa.

Após a configuração e adição das bibliotecas necessárias, é possível iniciar a implementação, que tem início na classe *TccApp.java*.

Nesta classe podemos alterar qualquer elemento gráfico da página e adicionar outros. Como não possuímos nenhum elemento gráfico, apenas criamos e adicionamos novos elementos no painel raiz da interface que representa o próprio elemento *body* da página.

Os elementos de interface no GWT-EXT são muitos, entre eles os simples e muito utilizados painéis, que funcionam como um *container* de objetos, botões, imagens e rótulos (*labels*), utilizados como caixa de texto. Entre os componentes mais avançados da biblioteca GWT-EXT estão: painéis de abas, formulários, *grids* editáveis, menus, árvores, *accordions* (um novo tipo de painel dinâmico), barras de progresso, entre outros.

Nossa interface é iniciada com um painel de *layout* complexo, que tem como principal característica separar seus elementos em as áreas: norte, sul, leste, oeste e centro. Isto permite criar *layouts* como mostra a Figura 19.



Figura 19 – Exemplo de layout complexo.

No layout complexo (nome dado pelos fabricantes da biblioteca) criado utilizamos as áreas norte, oeste e centro.

Na área norte, inserimos apenas uma imagem e um rótulo com o nome da aplicação. Já no setor oeste, inserimos um menu no estilo árvore com folhas que abrem novas abas. Estas abas são abertas no centro, que recebe um painel de abas.

A primeira aba é a aba inicial do programa, que não pode ser fechada. Nela foi adicionado um portal, pois ele permite a adição de *portlets* que são organizados em caixas, dando a impressão que várias janelas de conteúdo foram adicionadas e organizadas. Se houver necessidade, o desenvolvedor pode até permitir que os usuários fechem, minimizem ou movam estas janelas. A Figura 20 mostra a tela inicial do programa que possui um portal no setor central.

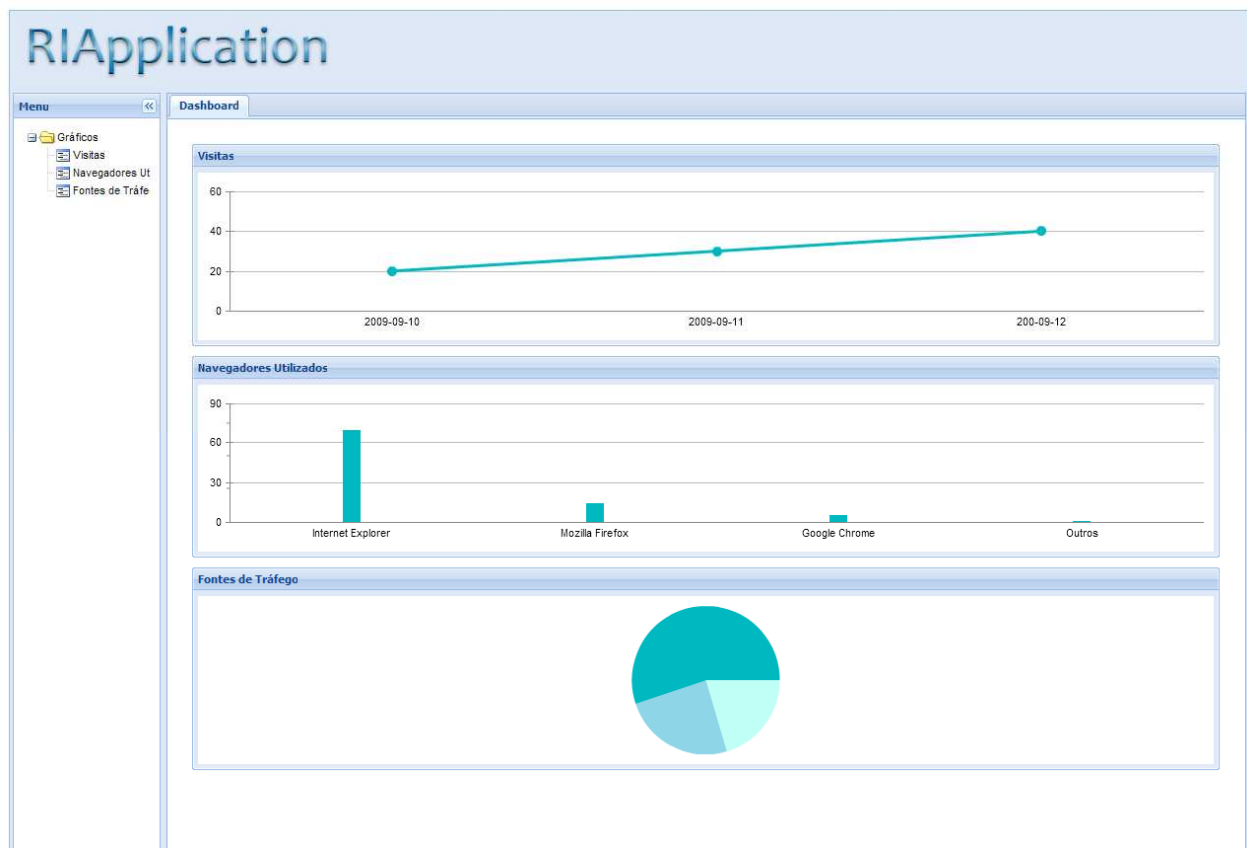


Figura 20 – Tela inicial da aplicação, o portal no centro com seus *portlets*.

Dentro desses *portlets* foram adicionados os gráficos.

Já as abas próprias para os gráficos foram feitas utilizando painéis que têm os gráficos inseridos. Esses painéis são abertos a partir do menu e podem ser fechados por um botão junto ao nome do painel.

Ao final do desenvolvimento pudemos levantar os pontos mais marcantes da tecnologia estudada. Logo no início, um desenvolvedor com conhecimentos em Java, se sente à vontade com a biblioteca, podendo manter seus padrões de projeto e estilo de programação. Isto acaba levando a um projeto bem organizado hierarquicamente, com classes pequenas e simples, que resulta em pacotes desacoplados e melhor manutenibilidade ao código.

A criação de componentes personalizados é extremamente fácil, permitindo que o

desenvolvedor estenda ou personalize alguma componente já existente sem nenhum sofrimento e de forma intuitiva.

O GWT em si não disponibiliza muitos componentes, pois se concentra em proporcionar uma plataforma de desenvolvimento de interfaces extremamente sólida. Esta falta de componentes é suprida pela vasta biblioteca GWT-EXT que entrega componentes ricos seguindo os padrões comerciais.

Este grande leque de componentes acaba gerando altos índices de produtividade. A implementação deste projeto levou em média 23 horas. Isto prova a rapidez da biblioteca, levando em conta que também foi necessário um aprendizado sobre a mesma.

Um dos maiores problemas encontrados no desenvolvimento foi a presença de *bugs* em alguns componentes. Um exemplo ocorre ao adicionar conteúdo num painel que já foi carregado no navegador. Se o desenvolvedor configurou o painel com algum layout, então o painel não irá adaptar seu tamanho para mostrar este conteúdo adicional.

Vale lembrar que todo este projeto foi feito por um desenvolvedor sem nenhuma experiência com a biblioteca GWT-EXT. É possível que as dificuldades encontradas possam ser sanadas com um pouco mais de experiência, mas não foram encontrados soluções nem casos similares, devido também à falta de suporte ao desenvolvedor.

4.2.4 Desenvolvimento FLEX

O desenvolvimento da aplicação Flex foi realizado em um período aproximado de 24 horas. Para tal, foi necessário o estudo de sua arquitetura, desde sua estrutura, instalação e suportes até utilização de eventos e funcionalidades. Segue abaixo as considerações sobre este

estudo e prática.

O Flex Builder proporciona em sua estrutura principal classes, componentes, arquivos CSS, arquivos ActionScript, arquivos MXML e uma aplicação principal, esta estrutura proporciona flexibilidade e performance. Os componentes dão uma visão de desacoplamento, facilidades de manutenção e reuso de código, assim como os arquivos ActionScript, e o desenvolvimento de componentes e sua integração com a aplicação é muito fácil de ser executada e deixa o código limpo.

A linguagem ActionScript se integra facilmente a MXML, permitindo navegar entre dados (XML), trocar informações e realizar *loops*. Um dos grandes trunfos encontrados nesta integração é a enorme facilidade em buscar, navegar, alterar e guardar dados na estrutura XML. Como exemplo pode-se visualizar a figura 21 demonstrando uma série de gráficos referentes a uma única requisição ao servidor, onde o Flex navega entre os dados do XML recebido da requisição ao servidor.



Figura 21 - Gráfico de visitantes de um determinado domínio, por ano, mês, semana e dia.

O Flex permite requisições via HTTPService a servidores ou Web Services. Com

facilidade configura-se uma requisição, e a cada *click* o usuário pode verificar se houve alterações nos dados, requisitando quantas forem às vezes necessárias.

O Flex permite também uma estrutura de estados, onde a partir de um botão em um componente a aplicação chame um novo estado filho, mostrando dependências entre os estados.

Indo a fundo no desenvolvimento, o Flex representou para nós autores uma ferramenta muito boa para criação de interfaces ricas, pois uma facilidade de desenvolvimento com complexidade gráfica, ou seja, com poucas e fáceis linhas de código montaram-se gráficos muito bonitos e representativos. Estes por sua vez apresentavam uma série de facilidades referente à chamada de eventos, atualização de dados, apresentação de informações, de modo a engrandecer ainda mais sua estrutura.

Como dificuldades encontradas, podemos citar:

1. A manipulação de dados com a estrutura `ArrayCollection` através de um XML;
2. Confusão entre estruturas como o `Panel` e `Canvas`;
3. Dificuldades ao encontrar erros referentes à ausência ou excesso de propriedades nas *tags*;
4. Dificuldades ao tentar utilizar a propriedade `itemRendered` dentro do `DataGrid`;

As dificuldades listadas acima são de pessoas que estão inicializando no mundo Flex, sendo a falta de entrosamento o maior obstáculo para um bom desenvolvimento. Após algum tempo de estudo e prática podemos observar o poder da linguagem, melhorando nosso desenvolvimento e engrandecendo nosso ponto de vista sobre a tecnologia.

No projeto utilizou-se o Flex Builder 3.3. Já está disponível a versão 3.4 beta do Flex Builder. Abaixo são listadas algumas das melhorias em relação versão 3.3 segundo Jon Rose:

1. Integração com o Adobe Catalyst (nova ferramenta de design da Adobe para

aplicações RIA sem escrita de código) - Promete mudar a forma de como os desenvolvedores e *designers* colaboram.

2. Arquitetura de componentes Spark – Nova biblioteca de componentes, que reimplementa os efeitos do Flex 4.
3. MXML 2209 – Atualização para suporte a diferentes comportamentos (núcleo, *skinning* e *layout*).
4. Melhorias nas ViewStates – Melhorias para simplificar e facilitar o uso.
5. Performance do Compilador – Considerado baixa, a Adobe vem juntando esforços para seu melhoramento.

Abaixo podemos visualizar o resultado do desenvolvimento do módulo Flex, a interface da aplicação final.

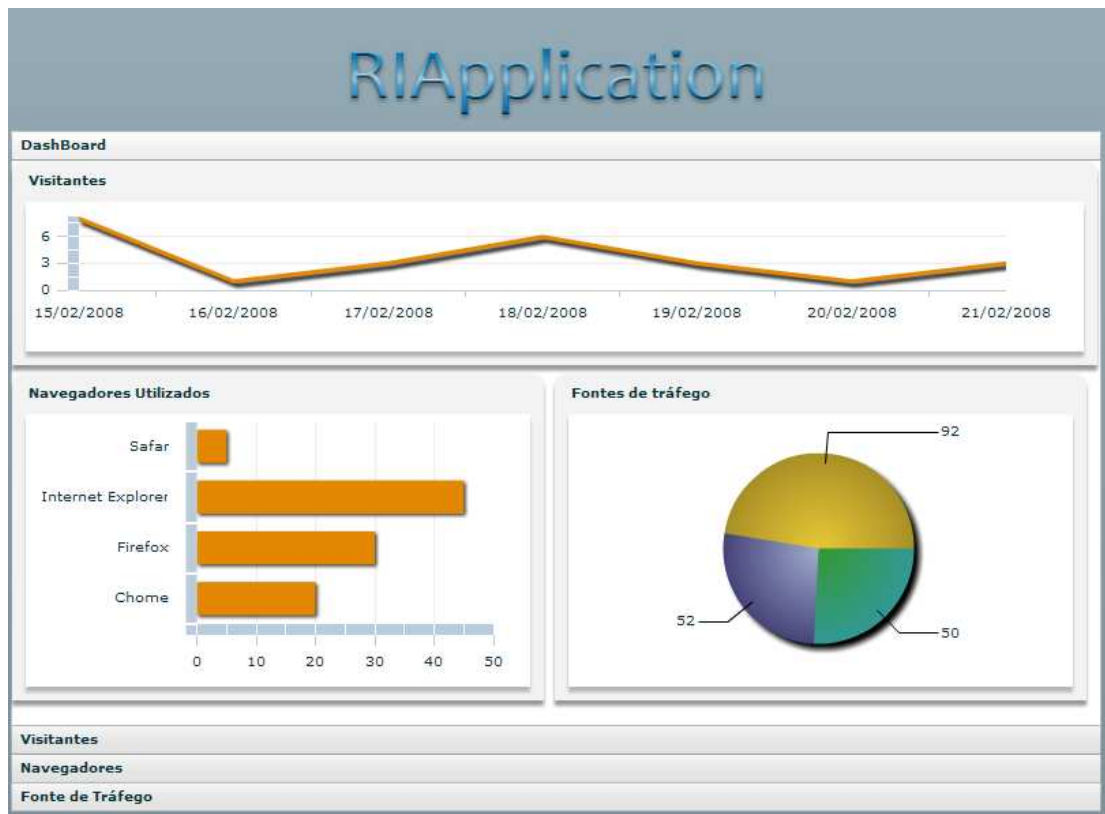


Figura 22 - Tela do módulo Adobe Flex .

5 CONCLUSÃO

Após o desenvolvimento deste projeto que utiliza as duas tecnologias, foram cruzadas suas características a fim de uma melhor compreensão destas. Além de um melhor entendimento do contexto de interfaces ricas para web. Segue abaixo esta comparação.

5.1 PLATAFORMA

O Adobe Flex utiliza o Flash Player como plataforma de execução, que está disponível para a maioria dos navegadores e é utilizado por 99% dos usuários da Web.

Já o GWT-EXT compila as classes da Interface para a linguagem JavaScript que é interpretada diretamente pelo navegador ou até, nos navegadores atuais as classes já em JavaScript são recompiladas para linguagem de máquina. Um dos problemas da utilização do JavaScript é que este pode ser desabilitado pelo usuário, ou seja, uma aplicação com GWT não funcionará.

Para comunicação com servidor o Adobe Flex executa requisições via HTTP GET, podendo comunicar-se com *web services* ou *servlets*, que podem retornar um XML como resposta.

No GWT, as classes Java compiladas em JavaScript fazem requisições via AJAX, que também utiliza o XML como formato para troca de dados, e o protocolo HTTP, que invoca a requisição em um *servlet* no servidor. Para o desenvolvedor este *servlet* é transparente, pois não precisa ser implementado, apenas estendido, deixando como tarefa do desenvolvedor apenas os métodos que tratam a requisição.

Como IDE (ambiente integrado para desenvolvimento de software) o Flex dispõe do Flex Builder, que entra em ação no desenvolvimento das classes Action Script e das interfaces MXML. Entretanto, a parte do servidor da aplicação precisa ser desenvolvida em outra IDE.

O Eclipse é o grande ajudante do desenvolvedor na plataforma GWT, pois permite a instalação do *plugin Google Plugin for Eclipse*, que atualmente suporta o *Google App Engine* (servidor de aplicações do Google) e o *Google Web Toolkit*. O próprio Eclipse é utilizado como suporte para desenvolvimento nas outras linguagens utilizadas na aplicação: Java, JavaScript, HTML e CSS.

Tanto o GWT quanto o Flex possuem como finalidade o desenvolvimento de aplicações WEB, mas também suportam a migração destas aplicações para o *desktop*, no caso do GWT através do GWT Desktop Framework, e no caso do Flex por intermédio da Adobe AIR *runtime*, com a qual a aplicação pode ser executada sem a necessidade de um navegador, entretanto a aplicação poderá utilizar a internet para comunicação com o servidor. Em relação a suporte de celulares e *smartphone* somente o Flex possui suporte, necessitando do aplicativo Flash Lite, que hoje suporta até a versão 8 do Flash Player, e a versão 2.0 do ActionScript.

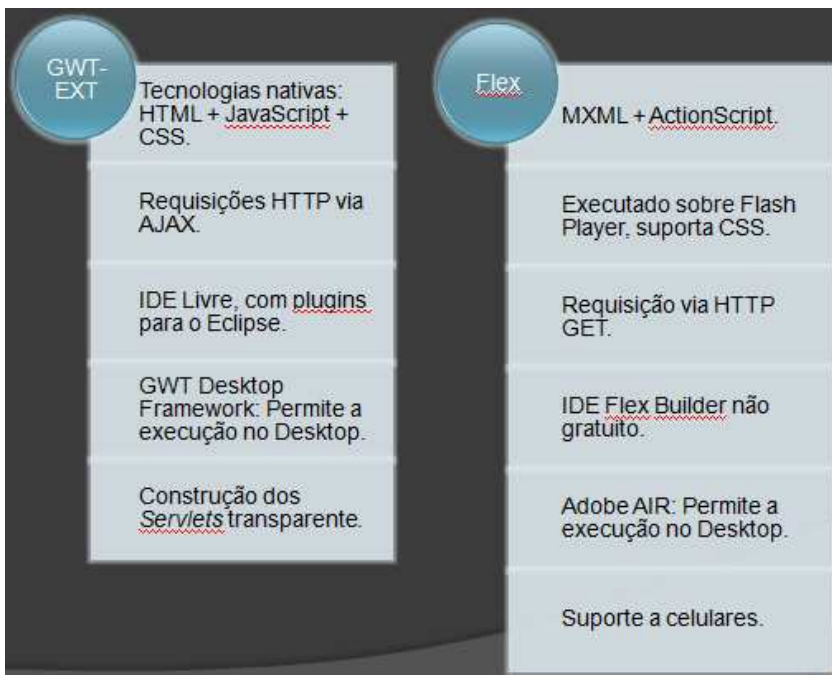


Figura 23 - Comparação entre a plataforma de desenvolvimento.

5.2 ESTRUTURA

Ambas as tecnologias organizam suas classes em pacotes, e devem ser importadas para poderem ser utilizadas em outras classes. Para deixar o código organizado e legível, cada classe poderá definir um componente.

No GWT também é feita a organização das classes da interface em um pacote enquanto as classes dos serviços são alojadas em outro. Usualmente as classes da interface ficam no pacote *client* e o pacote *server* guarda as classes de serviços.

Flex separa bem o trabalho do desenvolvedor e o trabalho do designer, pois este último trabalha com a linguagem MXML e se necessário, o CSS, que são responsáveis pelo layout, enquanto o desenvolvedor trabalha com Action Script, que é responsável pela camada de

comunicação com o servidor e pelo tratamento de eventos da interface. Já o GWT requer conhecimentos Java, HTML e em alguns casos, CSS e javascript, que dificultam o trabalho do designer, pois este precisa de conhecimentos em linguagem de programação (Java).



Figura 24 - Comparação entre a estrutura das tecnologias.

5.3 SUPORTE

Com mais tempo de mercado, o Adobe Flex conta com inúmeros fóruns de comunidades de desenvolvedores, onde trocam suas dúvidas em busca de soluções. Além dos tutoriais disponibilizados em seu site oficial, alguns até em vídeo, seus programadores contam também com um vasto leque de livros, artigos e publicações.

Em contrapartida, o GWT-EXT tem pouco tempo de vida, portanto não possui muitas publicações nem muitas comunidades. Estas poucas comunidades foram o suficiente para o desenvolvimento da aplicação, que teve como sua maior fonte de suporte a ótima galeria de exemplos disponível no site oficial do GWT-EXT.

Na etapa de desenvolvimento, que depende apenas da tecnologia GWT, foi possível

contar com uma maior quantidade de fontes de ajuda, além de ter uma grande comunidade em seu site oficial.



Figura 25 - Comparação entre o suporte aos usuários.

5.4 CÓDIGO

Nesta seção iremos mostrar as diferenças das tecnologias na questão de código nos arquivos-fonte. O GWT utiliza a linguagem de programação Java, logo, herda seus pontos positivos e também os negativos. Por ser uma linguagem imperativa, muitos desenvolvedores não vêem a linguagem Java como a linguagem ideal para programação de interfaces, já que a maioria das tecnologias de interfaces para a WEB conta com linguagens declarativas.

Já o Flex utiliza-se tanto de linguagem declarativa como imperativa, sendo o MXML declarativo, e o ActionScript imperativo, apresentando uma vantagem em relação ao GWT, por trazer em sua estrutura os dois tipos de linguagem.

Outra vantagem encontrada pelo Flex em relação ao GWT foi a abrangência de sua biblioteca, já que para o desenvolvimento dos gráficos da aplicação do GWT, foi necessário a instalação de uma biblioteca também disponibilizada pelo site oficial. Já na implementação em Flex, todas as funcionalidades foram desenvolvidas utilizando somente os recursos disponibilizados pelo pacote padrão.

Em relação à quantidade de linhas de código necessárias para implementar um mesmo componente, o Flex levou um vantagem com sua linguagem declarativa. Enquanto o componente de gráfico dos visitantes precisou de apenas 28 linhas de programação, o mesmo componente, implementado na plataforma GWT precisou de 44 linhas, o que representa um acréscimo de 57%.

Na chamada de serviços, os códigos em ambas as tecnologias são similares, pois chamam o serviço informando qual classe ou métodos devem ser invocados quando este serviço enviar uma resposta. Nas Figuras 26 a 31 são mostrados pequenos trechos dos códigos que requisitam os dados para o gráfico de visitantes.

```
@RemoteServiceRelativePath("visitorschartdataservice")  
public interface VisitorsChartDataService extends RemoteService {  
    String[][] getData();  
}
```

Figura 26 – Interface da classe de serviço que busca os dados do gráfico de visitantes no GWT.

```

/**
 * Construtor do gráfico
 */
public VisitorsDashboardWidget () {
    // seta alguns atributos do gráfico
    setTitle("Visitas");
    setDraggable(false);
    setCollapsible(false);

    // cria o objeto que irá tratar o retorno do serviço
    AsyncCallback<String[][]> callback = new VisitorsChartDataCallback<String[][]>();
    // cria o serviço
    VisitorsChartDataServiceAsync service = GWT
        .create(VisitorsChartDataService.class);
    // executa o serviço, passando como parâmetro o objeto que tratará seu
    // retorno
    service.getData(callback);
}

```

Figura 27 – Código que faz a requisição dos dados do gráfico de visitantes no GWT.

```

/**
 * Classe que trata o retorno do método que busca os dados do gráfico de
 * visitantes.
 */
private class VisitorsChartDataCallback<T> implements
    AsyncCallback<String[][]> {

    /**
     * Quando há uma falha no serviço, ou na chamada do serviço este método
     * é invocado.
     */
    public void onFailure(Throwable caught) {
        VisitorsDashboardWidget.this.add(new Label(
            "Ocorreu um erro ao buscar dados do gráfico.));
        VisitorsDashboardWidget.this.doLayout();
    }

    /**
     * Este método é invocado quando o serviço é executado com sucesso e
     * retorna os dados do gráfico.
     */
    public void onSuccess(String[][] result) {
        VisitorsChart chart = new VisitorsChart(result);
        chart.setHeight(200);
        VisitorsDashboardWidget.this.add(chart);
        VisitorsDashboardWidget.this.doLayout();
    }
}

```

Figura 28 - Classe que trata o retorno de uma requisição no GWT.


```

<mx:HTTPService
  <!-- Nome do serviço -->
  id="service"
  <!-- Declara a url que invoca o serviço -->
  url="http://localhost:8080/jsp/requisicao.jsp"
  <!-- Declara o método que será invocado o protocolo HTTP -->
  method="GET"
  <!-- Método invocado no script em caso de sucesso no envio e execução do serviço -->
  result="resultHandler(event)"
  <!-- Método invocado no script em caso de falha no envio e execução do serviço -->
  fault="faultHandler(event)" />
<!-- importa o script que contém os métodos utilizados neste componente -->
<mx:Script source="visitante.as" />

```

Figura 29 – Código que declara como deve ser invocado o serviço que busca os dados do gráfico de visitas no Flex.

```

// Função executada quando a aplicação é carregada,
// invoca o serviço que busca os dados do gráfico
// de visitantes.
function initApp():void{
    service.send();
}

```

Figura 30 – Função que invoca o serviço no servidor que busca os dados do gráfico de visitantes no Flex.

```

// Função que trata os dados retornados do serviço
// que busca os dados do gráfico de visitantes.
function resultHandler(event:ResultEvent):void{
    trace(ObjectUtil.toString(event.result));
    dadosInputVisitantes = event.result.acessos.acesso;
    dadosInputPeriodos = dadosInputVisitantes;
    flStatus = 0;
}

// Função que trata possíveis erros no serviço
// ou na invocação do mesmo.
function faultHandler(event:FaultEvent):void{
    Alert.show(ObjectUtil.toString(event.fault));
}

```

Figura 31 – Funções no Flex que tratam o retorno do serviço ou possíveis erros ocorridos.

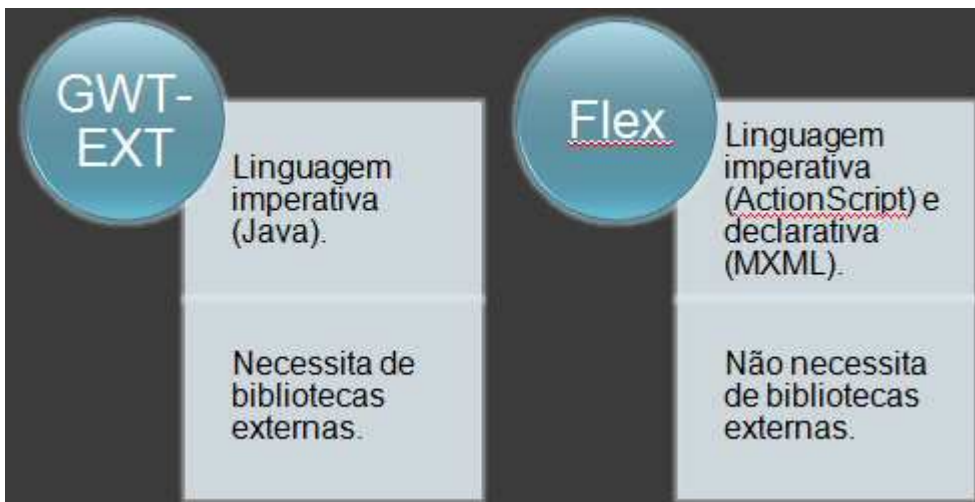


Figura 32 - Comparação entre código de programação.

5.5 DESEMPENHO

Para esta comparação foram levantados três indicadores de desempenho:

1. Compilação do código;
2. Inicialização da aplicação;
3. Velocidade de execução.

No primeiro item a aplicação em GWT obteve um maior tempo de compilação, tendo que compilar as classes Java para JavaScript marcando assim um tempo de 24,109 segundos, enquanto o Flex marcou um tempo de 0,38 segundos. Na comparação da velocidade de inicialização da aplicação o Flex novamente levou a melhor, marcando um tempo de 0,93 segundos e o GWT marcou um tempo de 1,61 segundos. Já na velocidade da aplicação obtivemos resultados semelhantes. Não encontramos nenhuma lentidão durante a aplicação.

Todos os testes foram realizados em mesmas condições num mesmo ambiente.

	GWT-EXT	<u>Flex</u>
Compilação do Código	24,109s	00,38s
Inicialização da Aplicação	00,93s	01,61s
Velocidade da Aplicação	Não foram observadas lentidões	Não foram observadas lentidões

Figura 33 - Comparação entre desempenho das aplicações.

5.6 DIFICULDADES

As dificuldades encontradas entre as tecnologias foram muito parecidas, desde integração com o *servlets* ao seu modo de funcionamento, porém após uma fase de familiarização com as tecnologias foi sentido que estas são de fácil aprendizado e muito sedutoras.

Também houveram algumas dificuldades devido a falhas na biblioteca GWT-EXT como, por exemplo, na adição de um elemento via requisição ao servidor em algum painel. Quando o painel tinha o tipo do *layout* definido, isso não funcionava (simplesmente não adicionava o elemento). Ao remover a linha de código que definia o tipo do seu *layout*, ele se adaptava com a adição do elemento e aumentava seu tamanho.

GWT-EXT	Flex
<ul style="list-style-type: none"> • <u>Bugs.</u> • Foi recentemente descontinuado. • Ágil e fácil para um programador Java. • Plataforma GWT sólida, sem <u>bugs.</u> 	<ul style="list-style-type: none"> • Tecnologia está em constante evolução. • Dificuldade no aprendizado da linguagem. • Bastante suporte. • Separa bem designers de programadores.

Figura 34 - Dificuldades encontradas no desenvolvimento.

5.7 CONSIDERAÇÕES FINAIS

A necessidade por aplicações para a WEB com interfaces ricas é real, entretanto, muitas empresas ainda têm receio em migrar suas aplicações com medo de estarem adotando a tecnologia errada ou que a mesma se tornará obsoleta rapidamente.

Visando preencher este espaço na área de software, inúmeras empresas estão aprimorando suas tecnologias na corrida para dominar o mercado com a sua solução. Neste quadro decidimos estudar algumas das tecnologias, selecionando duas para um estudo mais aprofundado.

A primeira tecnologia escolhida foi o Adobe Flex, por já ter um grande espaço no mercado, além de ter como plataforma o Adobe Flash Player, que está presente em 99% dos computadores com acesso à internet.

Foi notado que esta tecnologia está em constante evolução, como exemplo pode ser citado o lançado no início deste ano de uma nova versão do produto, este investimento da Adobe

gera credibilidade e confiança pela indústria de software, além de inúmeros fóruns, livros, estudos, cases, etc, sobre o tema.

Para nós desenvolvedores este fato é de grande importância visto que nossa primeira fonte de estudo, ou retirada de dúvidas geralmente é nesses locais (web, fóruns, livros, revistas).

Outro ponto importante deste investimento são as melhorias referentes à bugs, dificuldades encontradas, desempenho, novas facilidades, entre outras. Podemos citar como exemplo o trabalho da Adobe em desenvolver um ambiente separado de programação com o intuito de facilitar o trabalho dos designers. Ou ainda, o melhoramento da última versão em relação ao tempo de compilação.

No projeto Flex desenvolvido neste trabalho não foram encontrados bugs, mas sim algumas dificuldades na implementação, visto a não familiarização por parte do desenvolvedor, das linguagens utilizadas pelo Flex, como exemplo a utilização do debug ou até a identificação de erros referente à linguagem MXML. Porém as dificuldades foram desaparecendo, e em seu lugar surgiram idéias criativas, e até porque não, desejos de aplicações bonitas, robustas e facilitadoras.

O GWT-EXT foi a outra tecnologia escolhida, principalmente por suas duas maiores características: permitir a programação da interface utilizando a linguagem Java e utilizar tecnologias que são nativas do navegador WEB.

Baseado na biblioteca de terceiros Ext JS, o GWT-EXT foi recentemente descontinuado por problemas de licença, já que as novas versões do Ext JS foram lançadas com uma nova licença que não permitia que o GWT-EXT a utilizasse gratuitamente.

Um dos pontos frustrantes no desenvolvimento com o GWT-EXT foi o surgimento de diversas falhas (bugs) na biblioteca, e que se torna um problema ainda maior por conta da escassez de fontes de ajuda ao desenvolvedor.

Apesar disso, nossa experiência com o GWT-EXT provou que a plataforma GWT é

sólida e que, se combinada com outras bibliotecas, forma uma estrutura capaz de brigar com concorrentes já estabelecidos no mercado, permitindo o desenvolvimento de interfaces complexas desfrutando de altos níveis de produtividade.

Ambas as tecnologias utilizadas no desenvolvimento alcançaram um nível de produtividade equivalente, sendo que as duas marcaram um tempo de desenvolvimento em torno de 25 horas. Outro ponto que interfere no nível de produtividade é a quantidade de linhas de código, e mais uma vez, as tecnologias obtiveram resultados parecidos.

Todo caso de migração tem suas peculiaridades e deve ser estudado separadamente. Em geral, seguindo à risca o objetivo principal das interfaces RIA: desenvolver interfaces WEB da mesma maneira que são desenvolvidas interfaces *desktop*, o GWT é a tecnologia indicada pelos autores. A tecnologia realmente aproxima o desenvolvimento da interface à camada de negócios, pois ambas utilizam a mesma linguagem Java.

Além disso, desenvolvedores que já trabalham com esta linguagem têm uma curva de aprendizado rápida. Isso permite que empresas reaproveitem seus funcionários no desenvolvimento com o GWT.

Também, sua comunidade já conta com várias contribuições e está em constante crescimento pois vem contando com o apoio da enorme comunidade da linguagem Java. Aproveitando a flexibilidade do GWT e as contribuições da comunidade, já é possível a utilização de bibliotecas que expandem a plataforma como o GWT-EXT que foi utilizado neste projeto, e o Smart GWT. Ambas fornecem componentes ricos e são disponibilizadas pela comunidade GWT.

5.8 SUGESTÕES PARA TRABALHOS FUTUROS

5.8.1 Estudo de caso

Realizar um estudo de caso em uma empresa que deseja migrar sua tecnologia de interface para uma tecnologia RIA, buscando selecionar uma tecnologia que aumente a produtividade da empresa sem ter grande impacto na área de desenvolvimento e que tenha um desempenho de acordo com o perfil do sistema.

5.8.2 Ampliar o modelo da interface

Explorar o modelo da interface aplicada, ampliando o mesmo para uma comparação mais profunda das interfaces.

5.8.3 Aplicar o modelo em outras tecnologias

Aplicar o modelo em outras tecnologias, desenvolvendo outros módulos para a aplicação e comparando o desenvolvimento com as tecnologias já estudadas.

6 REFERÊNCIAS

ARAGAO JUNIOR, MAURICIO LINHARES. AJAX em Java com o Google Web Toolkit - AJAX rápido, fácil e puro Java com o Google Web Toolkit. Grupo de Usuários Java (GUJ), Junho. 2007. Disponível em: <<http://www.guj.com.br/article.show.logic?id=189#>>. Acesso em: 26 jun 2009.

BROWN, Charles E. The essencial guide to Flex 2 with ActionScript 3.0. Editora FriendSof. 2007.

DOEDERLEIN, OSVALDO PINALI. A guerra do RIA – Por dentro do TraceMonkey, Chrome / V8. Java Magazine, Ed. 63, p. 45-56. Novembro. 2008.

FLASH Player penetration. Estados Unidos da América, 2009. Disponível em: <http://www.adobe.com/products/player_census/flashplayer/> . Acesso em 30 nov. 2009.

FREDERICK, SHEA; RAMSAY, COLIN; BLADES, STEVE ‘CUTTER’. Learning Ext JS. Birmingham: Packt Publishing, 2008. 299 p.

JAYDSON NASCIMENTO GOMES, RODRIGO PRESTES MACHADO. RichBlocks – Um framework para implantar interfaces RIA em sistemas web. Rio Grande do Sul. Curso de Análise e Desenvolvimento de Sistemas. Faculdade de Tecnologia SENAC RS (FATEC/RS).

MARTINS, ANA CAROLINA COSTA. Projeto de interfaces gráficas para web [trabalho de conclusão de curso]. Juiz de Fora: Universidade Federal de Juiz de Fora. Curso de Ciência da Computação. Departamento de Ciência da Computação, 2007.

MARTINS, RICARDO ALEXANDRE G. C.; RAMALHO, JOSÉ CARLOS; HENRIQUES, PEDRO RANGEL. Estudo comparativo de diferentes linguagens de interfaces baseadas em XML. Portugal. Departamento de Informática — Universidade do Minho.

MONTEIRO, ANDRÉ LUIZ. RIA com Open Laszlo – Construindo Rich Internet Applications com Flash e Java. Java Magazine, Ed. 39, p. 70-74. Abril. 2008.

MORONEY, LAURENCE. Introducing Microsoft Silverlight 1.0. Washington: Microsoft Press, 2008. 141 p.

NELSON ROGÉRIO VICTORAZZI. RIA – Rich Internet Applications [Monografia para obtenção de grau]. Curso de especialização em WEB e Sistemas de Informação. Universidade Federal do Rio Grande do Sul, 2007.

RODRIGUES, DOUGLAS JOSÉ SOARES. Ajax com Google Web Toolkit – Escrevendo aplicações web altamente interativas em Java. Java Magazine, Ed. 38, p. 64-67. Maio. 2008.

ROSE, JON. As 10 Maiores Mudanças no Flex 4. Comunidade Online InfoQ, Junho 2007. Disponível em <<http://www.infoq.com/br/articles/top-10-flex4-changes>>. Acesso em: 10 de Outubro de 2009.

SLENDER, GRANT K.. Developing with Ext GWT: Enterprise RIA Development. Berkeley: Apress, 2009. 140p.

TERRACINI, FABIO; MARTINELLI, RAFAEL. Adobe Flex 2 – Crie aplicações para a web com interface rica,amigável e integrada a aplicações Java existentes. Mundojava, Curitiba, p. 52-63.

UEHARA, AUGUSTO. JavaFX – Interfaces de alto impacto em Java. Mundojava, Curitiba, n. 24, p. 40-46, Julho/Agosto. 2007.

7 ANEXOS E APÊNDICES

7.1 ARTIGO

ADOBE FLEX X GWT-EXT. Tecnologias RIA para aplicações Java com desenvolvimento no Eclipse.

Gustavo S. Schneider¹, Tiago Braun Corrêa¹

¹Instituto de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC - Brasil
{gss, braun87}@inf.ufsc.br

Resumo. *Com o aumento da importância da Web como veículo de comunicação e o boom da Web 2.0 houve uma popularização de muitas tecnologias de interface rica (RIA), além da criação de novas.*

Algumas delas não apenas deixam as aplicações mais modernas e bonitas, como também adicionam novas funcionalidades e facilidades, tanto para o usuário quanto para os desenvolvedores de software, que se vêem obrigados a adequar seus produtos a essas novas tecnologias.

Nesse contexto, surge a dúvida de qual tecnologia se adequa melhor a um determinado produto, haja vista que existem semelhanças entre as tecnologias, porém há também muitas características divergentes que diferenciam cada uma das tecnologias.

Neste trabalho são estudadas algumas dessas tecnologias e são comparados com mais detalhes o Adobe Flex e o GWT-EXT tendo como base o desenvolvimento de uma aplicação.

Tanto o desenvolvimento quanto a comparação tem por finalidade dar base a uma escolha da tecnologia mais indicada para desenvolvimento de interfaces ricas para WEB utilizando-se da linguagem Java tendo como IDE de apoio o Eclipse.

Abstract. *With the raising importance of the Web as a mean of communication and the boom of Web 2.0 there has been a popularization of many rich interfaces technologies and the creation of new ones.*

Some of these not only make the application prettier and more modern, but also add more functionalities and facilities for the user as well as for the software developers. However, developers feel forced to adapt their products to these new technologies.

In this context one may question which technology is better for a certain product, as there are many similar characteristics among them, but also many distinct ones that

differentiate one technology from another.

Using the development process as a base, this paper studies some of these technologies and thoroughly compares Adobe Flex and GWT-EXT.

Both the development and the comparison are intended to provide support for choosing the most appropriate technology when developing rich interfaces for WEB using the Java language and Eclipse as an IDE.

2 Introdução

Com o aumento da importância da internet como veículo de comunicação surge também a necessidade de melhorar a experiência do usuário na internet. Novas tecnologias de interface são desenvolvidas e utilizadas. Algumas delas não apenas deixam a interface mais moderna ou bonita, mas também adicionam novas funcionalidades e facilidades, tanto para o usuário quanto para o programador da mesma.

A importância da interface é enfatizada por Ana Carolina Costa Martins (2007), “Uma boa interface pode ser determinante na aceitação de determinado produto, uma vez que a execução das tarefas da aplicação depende da forma que os elementos são apresentados ao usuário”.

O boom da Web 2.0 levou à popularização de muitas tecnologias além da criação de novas, o que acaba gerando alguns receios para o programador que gostaria de atualizar a interface de seu sistema: qual é a tecnologia que deve ser adotada analisando desempenho da aplicação, suporte aos usuários, plataforma de desenvolvimento, dificuldades de implementação, código de programação e estrutura de projeto.

A necessidade de integrar ao sistema uma interface de ponta surge com a competitividade do mercado. Neste contexto, empresas de desenvolvimento de software vêm adotando novas tecnologias RIAs em busca da satisfação de seus clientes, sua permanência no mercado e a longo prazo um aumento na cadeia de clientes.

Porém a adoção de uma nova tecnologia interfere no ambiente da empresa tanto na estrutura operacional, tendo que redefinir seus processos de desenvolvimento de software, tanto no ambiente organizacional, com novas áreas, novas funções e contratações, além de treinamento. Sendo a escolha da adoção de uma nova tecnologia um importante mecanismo para o sucesso da organização.

Visto isso, este trabalho tem como objetivo comparar duas tecnologias RIAs promissoras, o Adobe Flex e GWT-EXT, nos quesitos: Plataforma de desenvolvimento; Estruturas de projeto;

Suporte dado aos usuários; Código de programação; Desempenho das tecnologias; Dificuldades de implementação encontradas.

3 Surgimento de RIA

Segundo Doedelein a “incapacidade dos padrões Web (como HTML e CSS) para implementar funcionalidades de GUI mais avançadas, incluindo suporte rico a mídias em geral, o insuficiente respeito a estes padrões, mesmo quando estes deveriam ser suficientes e o fraco desempenho da linguagem JavaScript”, são os fatores principais para o surgimento das tecnologias RIA.

(...) O conceito de Rich Internet Application(RIA) é uma resposta a este problema. Permitindo que o usuário interaja com a aplicação como se esta fosse um sistema desktop tradicional, o RIA baseia-se na premissa de rodar na maquina do usuário todo o processamento de interface gráfica, deixando para o lado do servidor o processamento da lógica de negócio.(MONTEIRO, 2008)

As aplicações que se utilizam do RIA “se assemelham as aplicações desktop, fazendo com que seu uso seja mais fácil, oferecendo uma interface mais rica aos usuários”. (LOOSLEY, 2006). Segundo BRIDEE, RIA é mais que uma tecnologia, é um conceito, reforçando que RIA é uma combinação de interatividade e funcionalidade do desktop com a flexibilidade e abrangência da Web.

Como ferramentas RIA pode-se citar: Adobe Flash, Flex e Air, Microsoft SilverLight, GoogleWebToolkit, JavaFX, OpenLazlo, e Frameworks baseados em Ajax.

4 Google Web Toolkit

Desenvolvido dentro do Google e liberado pelos seus desenvolvedores como software livre o GWT pode ser escrito totalmente com código Java, utilizando-se de ambientes de desenvolvimento integrados de alta qualidade, tipagem estática e debug ativo, tendo um compilador especial que transforma o código Java em código JavaScript, para que ele possa executar no navegador cliente.

Vejamos alguns pontos importantes no uso desta ferramenta segundo Maurício Linhares de Aragão Junior (2007).

Criação do projeto: Executar o utilitário `projectCreator -ant GWTTutorial -eclipse GWTTutorial`, esse comando vai instruir a ferramenta a criar um “buildfile” do Ant com o nome

GWTTutorial.ant.xml e os arquivos de configuração necessários para transformar esse projeto em um projeto do Eclipse. Depois de criados os arquivos, você pode importar esse projeto para dentro do Eclipse utilizando a importação de projetos.

Padrão de design: Pacote criado deve, preferivelmente, terminar com o nome “client”, porque esse é o pacote onde vão estar as classes e interfaces que vão ser transformadas em JavaScript para executar no cliente. Além desse pacote, o GWT cria um pacote “public” onde vão estar disponíveis os arquivos de recursos para a aplicação (como os arquivos HTML, CSS e imagens). O outro pacote que pode existir é o “server” que guarda as classes que não são transformadas em JavaScript e são utilizadas através da API de invocação de serviços remotos do GWT. Os nomes das pastas são configuráveis, mas é uma boa prática manter os nomes padrão para evitar dores de cabeça futuras com configurações.

Invocação remota de serviços: A aplicação JavaScript que executa no cliente faz uma invocação a um serviço implementado no servidor (através de um Servlet) que pode então se utilizar de toda a expressividade e APIs do Java para retornar os dados para o cliente, como o JDBC, acesso a arquivos, EJB/Hibernate e outros

Douglas José Soares Rodrigues 2008 conclui que “o GWT mostra-se um framework extremamente eficiente e bem escrito, além de ser uma solução robusta para o desenvolvimento AJAX com o suporte de uma empresa renomada”.

5 Ext JS

Ext provê uma interface rica, fácil de usar, muito parecidas como as que você encontraria em uma aplicação desktop. Isto permite que desenvolvedores WEB concentrem na funcionalidade da aplicação web ao invés de suas ressalvas técnicas (FREDERICK; RAMSAY; BLADES).

Uma grande biblioteca JavaScript poderia ser um resumo do Ext, mas ele vai além disso, não só oferece um vasto leque de componentes como painéis, grids, forms, menus, janelas, abas, botões, etc., mas também dá outra perspectiva à aplicação com seus três pontos fortes:

- Ampla compatibilidade em navegadores: as problemáticas diferenças na interpretação de código JavaScript e CSS entre os navegadores são tratadas automaticamente pela biblioteca que dá suporte aos principais navegadores: Internet Explorer 6+, Firefox 1.5+ (PC, MAC), Safari 2+. Opera 9+ (PC, MAC).

- Interfaces baseadas em eventos: um evento poderia ser uma ação feita pelo usuário, como o clique de um botão ou gerado pela própria aplicação, como o carregamento da aplicação ou o retorno de uma requisição de dados. São estes eventos que invocam as classes implementadas pelo desenvolvedor.
- AJAX: A biblioteca utiliza AJAX para a comunicação com o servidor de maneira simples e transparente. Isso permite, por exemplo, carregar um *grid* enquanto um *form* é preenchido.

6 GWT-EXT

Desde o lançamento do Google Web Toolkit (GWT) e frameworks de widgets RIA relacionados como o Ext GWT, o desenvolvimento de RIA agora é muito mais fácil e flexível. Então, se você consegue construir aplicações Java, você pode construir rapidamente uma aplicação rica para internet de classe empresarial (SLENDER, GRANT).

Baseado na biblioteca JavaScript Ext JS o GWT-EXT é uma poderosa biblioteca de *widgets* como *grids*, painéis, *forms*, etc., para a plataforma GWT.

Basicamente o GWT-EXT é uma migração dos componentes da biblioteca EXT JS para a plataforma GWT, o que permite a utilização da linguagem de programação Java no lugar da linguagem JavaScript, que é utilizada no EXT JS.

Apesar de ter apresentado uma boa recepção pela comunidade de programadores WEB, o projeto GWT-EXT não terá continuação. No fórum oficial da comunidade a equipe do GWT-EXT está aconselhando seus usuários a migrarem para outra biblioteca RIA-GWT como a SmartGWT. Isto se deve ao fato da licença do projeto Ext JS ter sido alterada, que até a versão 2.0.2 mantinha a licença LGPL.

7 Adobe Flex 2

Em 2004 a Macromedia Flex, hoje conhecida como Adobe Flex, divulgou o lançamento do produto Flex, hoje com cinco anos de vida e que já atinge um alto grau de utilização pelas empresas de sistemas Web.

Sua proposta de página única desbanca o estilo antigo de páginas em HTML com links, onde a navegação consiste em página por página, com inúmeras requisições e possibilidades de erros como página não encontrada. Flex destaca-se pela a notável diferença de usabilidade entre

as tecnologias usualmente empregadas para desenvolvimento de aplicações Web.

Em sua estrutura a Adobe escolheu integrar-se com a IDE Eclipse, visto que os desenvolvedores possuem afinidade com esta, facilitando assim sua adaptação. Para tal basta baixar o plug-in do Flex Builder, que requer licença.

O runtime do Flex é o Flash, requerendo a instalação de no mínimo o Flash 9 player . Apesar de rodar em Flash, o Flex não se caracteriza como um programa de animação.

Para desenvolvimento das aplicações a Adobe desenvolveu o Flex Builder, que provê um ambiente no qual o desenvolvedor e o designer trabalham juntos: o desenvolvedor com a linguagem ActionScript ,e o designer com a linguagem MXML.

A linguagem MXML, derivada do XML, fornece suporte a estilos CSS (Cascading Style Sheets), facilidades em alternar cores e navegar entre diferentes funcionalidades. Com o Flex Builder o designer visualiza o código da aplicação e seu design, podendo redimensionar e editar o código MXML.

O adobe Flex é uma solução completa para criar e fornecer aplicações ricas, robustas, interativas e que possibilizem uma interface mais anigável e intuitiva para o usuário. Martinelli e Terracini.

Composto pelos seguintes itens:

Adobe Flex 2 SDK - Conjunto de classes (framework), compilador, debbuger, duas linguagens de programação MXML e Actionsript além de incluir códigos fontes do conjunto de classes para customização pelo desenvolvedor.

Adobe Flex Builder 2 (IDE).

Adobe Flex Data Services 2 - Serviço de mensagem que roda em um servidor JEE compatível ou em contênier Servlet.

Adobe Flex Charting 2 - Funcionalidade que facilita a visualização de um conjunto de dados no formato de gráficos de duas dimensões.

O modelo de aplicativo pode ser representado pelo modo como ele é executado no servidor. Sendo estes: Client-side Only Applications (executada no cliente não utilizando recursos do servidor), Data Acess With HTTPServices and WebServices (comunicação com o servidor através de chamadas HTTP ou utilizando WEB services) e Data Acess With Flex Data Services

(vantagens no acesso ao servidor, tais como, sincronização, segurança e comunicação). Segundo Martinelli e Terracini “a forma mais simples é ler em XMLs via conexão HTTP, porém também é possível utilizar o padrão SOAP para conectar a Web Services.”

8 Aplicação

O tipo de aplicação foi um “Dashboard” para a visualização de gráficos e relatórios de acesso a sites de internet, inspirado no Google Analytics. Este “Dashboard” ou painel é preenchido com vários quadrados que formam áreas para diferentes funcionalidades que mostram estatísticas.

A aplicação é executada sobre o servidor de aplicação JBoss e ao ser acessada, o usuário faz um contato com o contexto WEB, o pacote “cliente” da aplicação, que executa o código da tecnologia acessada. Este código se comunica com o servidor via requisições http aos *servlets*, localizados nos módulos GWT ou Flex.

A IDE utilizada no projeto foi o Eclipse. A configuração da aplicação no servidor incluiu estruturar a mesma no padrão J2EE, além de criar os arquivos de configuração da aplicação no JBoss. A figura 1 representa a estrutura final da aplicação:

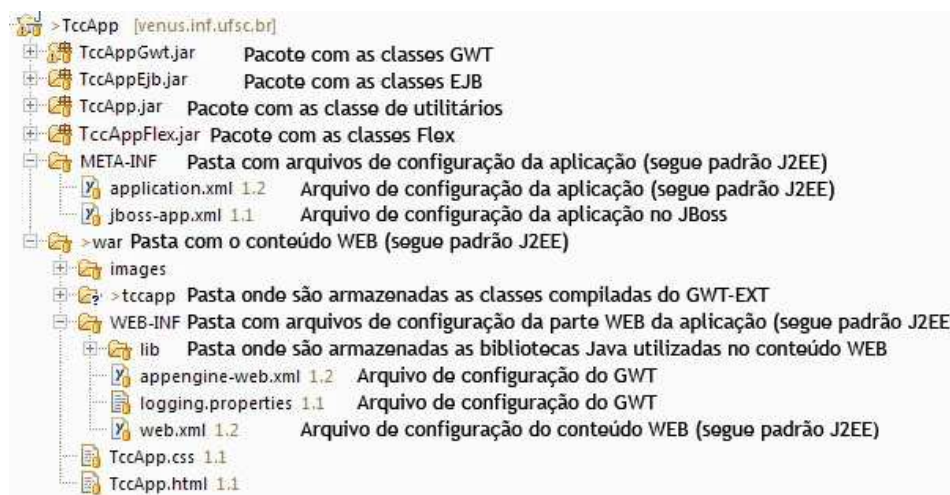


Figura 35 - Estrutura da aplicação, baseada no padrão J2EE.

9 Desenvolvimento

O desenvolvimento do módulo GWT da aplicação utilizou-se da biblioteca GWT-EXT rodando em uma plataforma GWT. Esta plataforma trabalha ao todo com quatro tecnologias:

Java: utilizada na maior parte do código, na construção da camada de negócios, da camada de acesso aos dados e também da camada de interface. Esta última, porém, também utiliza outras

linguagens.

HTML: é o *container* da camada de interface. No HTML é inserido o código JavaScript resultante da compilação do módulo GWT que se comunica com a camada de negócios. Também invoca a utilização da tecnologia CSS.

JavaScript: em geral é gerado no processo de compilação do módulo GWT, entretanto ainda permite que o desenvolvedor crie e utilize suas próprias funções, se houver necessidade. No caso da aplicação, não foi implementada nenhuma função JavaScript. Este também é o módulo responsável pela comunicação com o servidor, via requisições XML (AJAX), invoca métodos implementados em Java, localizados no módulo GWT, na camada de negócios.

CSS: aplica estilos no *container* HTML, cores, posicionamento, tamanhos. Não altera o que o usuário final vê na interface, mas sim como ele a vê.

Já o desenvolvimento do módulo Flex foi estruturado a partir de componentes, módulos, arquivos CSS, arquivos ActionScript, e uma aplicação principal. A aplicação principal é formada pela linguagem MXML, que integram componentes e módulos a sua estrutura. Para comunicação com os servelts é utilizado a linguagem JavaScript que se integra muito bem a linguagem MXML. Todos esses arquivos são compilados gerando um único arquivo no formato. swf que é integrado ao projeto Java no Eclipse.

A grande diferença no desenvolvimento dos módulos é que com o GWT o desenvolvimento foi realizado todo na IDE Eclipse enquanto no Flex o desenvolvimento é realizado em conjunto Eclipse e Flex Builder.

A figura 2 descreve os principais pontos do processo de desenvolvimento da aplicação, que foi dividida em duas etapas, configuração e implementação. A fase de configuração compreendeu-se na preparação do ambiente de desenvolvimento, Eclipse e Flex Builder, importação da biblioteca GWT-EXT, para o módulo do GWT e por fim configuração do Servidor, no caso Jboss. A fase de implementação iniciou com a criação do layout, seguido do desenvolvimento do dashboard e por fim a construção dos gráficos e integração dos módulos com o servidor.

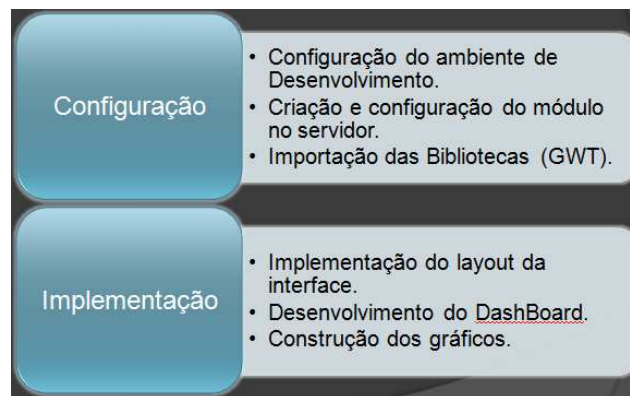


Figura 36 - Processo de desenvolvimento da aplicação.

10 Conclusão

Após o desenvolvimento deste projeto que utiliza as duas tecnologias, foram cruzadas suas características a fim de uma melhor compreensão destas. Além de um melhor entendimento do contexto de interfaces ricas para web. Segue abaixo esta comparação.

Plataforma

A figura 3 abaixo exemplifica as principais características de cada tecnologia no quesito plataforma. Conforme podemos observar as linguagens utilizadas são divergentes apesar das duas aceitarem o uso da tecnologia CSS, não só as linguagens como o tipo de requisição ao servidores, IDE de apoio, runtimes divergem. Por fim somente o Flex oferece suporte a celulares.

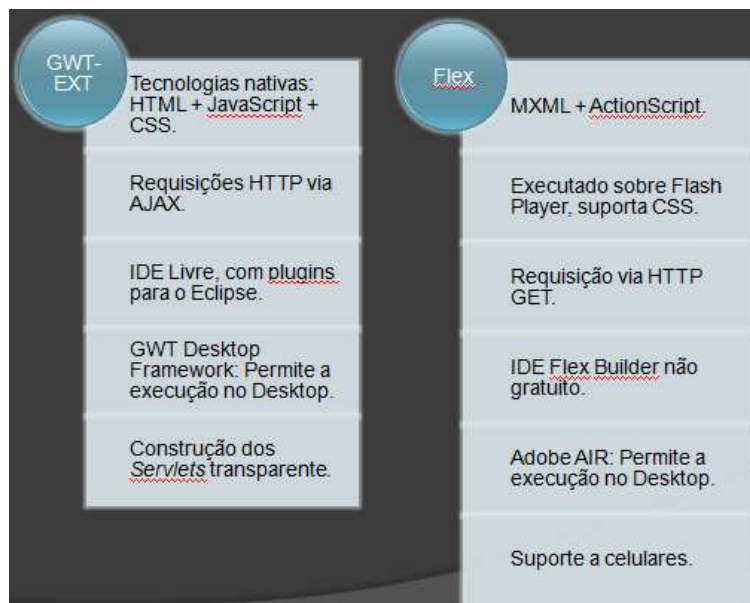


Figura 37 - Comparação entre a plataforma de desenvolvimento.

Estrutura

O GWT-EXT trabalha quase na totalidade com a linguagem Java, não diferenciando as linguagens trabalhadas por designers e programadores, ao contrário do Adobe Flex onde os designers utilizam a linguagem estática MXML e os desenvolvedores trabalham com a linguagem dinâmica ActionScript e sua integração com os servlets.

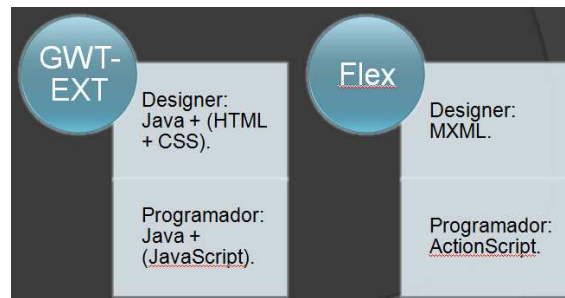


Figura 38 - Comparação entre a estrutura das tecnologias..

Suporte

No quesito suporte o Adobe Flex trouxe maiores opções de suporte, com inúmeras publicações de artigos e livros. Já o GWT ainda é prematuro o número de publicações. Porém as duas tecnologias apresentam boas informações em seus WebSites. A figura 5 abaixo demonstra exemplifica o suporte dado aos usuários.



Figura 39 - Comparação entre o suporte aos usuários.

Código

Em relação ao código as duas linguagens mostraram ser sucintas e poderosas criando interfaces bonitas e funcionais. O GWT precisou importar bibliotecas internas para construção dos gráficos, ao contrário do Flex.



Figura 40 - Comparação entre código de programação.

Desempenho

Para comparação do desempenho foi levantado 3 indicadores, tempo de compilação do código; inicialização da aplicação e velocidade da aplicação. A figura 7 mostra o resultado do estudo.

	GWT-EXT	Flex
Compilação do Código	24,109s	00,38s
Inicialização da Aplicação	00,93s	01,61s
Velocidade da Aplicação	Não foram observadas lentidões	Não foram observadas lentidões

Figura 41 - Comparação entre desempenho das aplicações.

Dificuldades

A figura 8 relata as principais dificuldades encontradas na implementação e utilização das tecnologias GWT e Flex.

GWT-EXT	Flex
<ul style="list-style-type: none"> • <u>Bugs</u>. • Foi recentemente descontinuado. • Ágil e fácil para um programador Java. • Plataforma GWT sólida, sem <u>bugs</u>. 	<ul style="list-style-type: none"> • Tecnologia está em constante evolução. • Dificuldade no aprendizado da linguagem. • Bastante suporte. • Separa bem designers de programadores.

Figura 42 - Dificuldades encontradas no desenvolvimento.

Considerações Finais

Todo caso de migração tem suas peculiaridades e deve ser estudado separadamente. Em geral, seguindo à risca o objetivo principal das interfaces RIA: desenvolver interfaces WEB da mesma maneira que são desenvolvidas interfaces desktop, o GWT é a tecnologia indicada pelos

autores. A tecnologia realmente aproxima o desenvolvimento da interface à camada de negócios, pois ambas utilizam a mesma linguagem Java.

Além disso, desenvolvedores que já trabalham com esta linguagem têm uma curva de aprendizado rápida. Isso permite que empresas reaproveitem seus funcionários no desenvolvimento com o GWT.

Também, sua comunidade já conta com várias contribuições e está em constante crescimento pois vem contando com o apoio da enorme comunidade da linguagem Java. Aproveitando a flexibilidade do GWT e as contribuições da comunidade, já é possível a utilização de bibliotecas que expandem a plataforma como o GWT-EXT que foi utilizado neste projeto, e o Smart GWT. Ambas fornecem componentes ricos e são disponibilizadas pela comunidade GWT.

11 Referências

- Aragao Junior, Mauricio Linhares. AJAX em Java com o Google Web Toolkit - AJAX rápido, fácil e puro Java com o Google Web Toolkit. Grupo de Usuários Java (GUJ), Junho. 2007. Disponível em: <<http://www.guj.com.br/article.show.logic?id=189#>>. Acesso em: 26 jun 2009.
- Brown, Charles E. The essencial guide to Flex 2 with ActionScript 3.0. Editora FriendSof. 2007.
- Doederlein, Osvaldo Pinali. A guerra do RIA – Por dentro do TraceMonkey, Chrome / V8. Java Magazine, Ed. 63, p. 45-56. Novembro. 2008.
- Flash Player penetration. Estados Unidos da América, 2009. Disponível em: <http://www.adobe.com/products/player_census/flashplayer/> . Acesso em 30 nov. 2009.
- Frederick, Shea; Ramsay, Colin; Blades, Steve ‘Cutter’. Learning Ext JS. Birmingham: Packt Publishing, 2008. 299 p.
- Martins, Ana Carolina Costa. Projeto de interfaces gráficas para web [trabalho de conclusão de curso]. Juiz de Fora: Universidade Federal de Juiz de Fora. Curso de Ciência da Computação. Departamento de Ciência da Computação, 2007.
- Monteiro, André Luiz. RIA com Open Laszlo – Construindo Rich Internet Applications com Flash e Java. Java Magazine, Ed. 39, p. 70-74. Abril. 2008.
- Nelson Rogério Victorazzi. RIA – Rich Internet Applications [Monografia para obtenção de grau]. Curso de especialização em WEB e Sistemas de Informação. Universidade Federal do Rio Grande do Sul, 2007.
- Rodrigues, Douglas José Soares. Ajax com Google Web Toolkit – Escrevendo aplicações web altamente interativas em Java. Java Magazine, Ed. 38, p. 64-67. Maio. 2008.
- Rose, Jon. As 10 Maiores Mudanças no Flex 4. Comunidade Online InfoQ, Junho 2007. Disponível em <<http://www.infoq.com/br/articles/top-10-flex4-changes>>. Acesso em: 10 de Outubro de 2009.

Slender, Grant K. Developing with Ext GWT: Enterprise RIA Development. Berkeley: Apress, 2009. 140p.

Terracini, Fabio; Martinelli, Rafael. Adobe Flex 2 – Crie aplicações para a web com interface rica, amigável e integrada a aplicações Java existentes. Mundojava, Curitiba, p. 52-63.

Uehara, Augusto. (2007) “JavaFX – Interfaces de alto impacto em Java. Mundojava”, Curitiba, n. 24, p. 40-46, Julho/Agosto.

7.2 CÓDIGO FONTE DO MÓDULO GWT

7.2.1 Arquivo de configuração TccApp.gwt.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE module PUBLIC "-//Google Inc.//DTD Google Web Toolkit 1.7.0//EN"
"http://google-web-toolkit.googlecode.com/svn/tags/1.7.0/distro-
source/core/src/gwt-module.dtd">
<module rename-to='tccapp'>
  <!-- Inherit the core Web Toolkit stuff. -->
  <inherits name='com.google.gwt.user.User' />

  <!-- Inherit the default GWT style sheet. You can change -->
  <!-- the theme of your GWT application by uncommenting -->
  <!-- any one of the following lines. -->
  <!--
    <inherits name='com.google.gwt.user.theme.standard.Standard' />
    DEFAULT
  -->
  <!-- <inherits name='com.google.gwt.user.theme.chrome.Chrome' /> -->
  <!-- <inherits name='com.google.gwt.user.theme.dark.Dark' /> -->

  <!-- Other module inherits -->
  <!-- Inherit the core GWT-Ext Toolkit stuff. -->
  <!-- GWT-EXT - Importa a biblioteca do GWT-Ext Toolkit. -->
  <inherits name='com.gwttext.GwtExt' />
  <inherits name='com.gwttext.Charts' />

  <!-- Specify the app entry point class. -->
  <entry-point class='br.ufsc.inf.tccapp.gwt.client.TccApp' />

  <!-- GWT-EXT - Arquivos de estilo e scripts da biblioteca EXT -->
  <stylesheet src="js/ext/resources/css/ext-all.css" />
  <script src="js/ext/adapter/ext/ext-base.js" />
  <script src="js/ext/ext-all.js" />
</module>
```

7.2.2 Arquivo de configuração jdoconfig.xml

```
<?xml version="1.0" encoding="utf-8"?>
<jdoconfig xmlns="http://java.sun.com/xml/ns/jdo/jdoconfig"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://java.sun.com/xml/ns/jdo/jdoconfig">

  <persistence-manager-factory name="transactions-optional">
    <property name="javax.jdo.PersistenceManagerFactoryClass"
value="org.datanucleus.store.appengine.jdo.DatastoreJDOPersistenceManagerFacto
ry"/>
    <property name="javax.jdo.option.ConnectionURL" value="appengine"/>
    <property name="javax.jdo.option.NontransactionalRead" value="true"/>
    <property name="javax.jdo.option.NontransactionalWrite" value="true"/>
    <property name="javax.jdo.option.RetainValues" value="true"/>
    <property name="datanucleus.appengine.autoCreateDatastoreTxns"
value="true"/>
  </persistence-manager-factory>
</jdoconfig>
```

7.2.3 Arquivo de configuração log4j.properties

```
# A default log4j configuration for log4j users.
#
# To use this configuration, deploy it into your application's WEB-INF/classes
# directory. You are also encouraged to edit it as you like.

# Configure the console as our one appender
log4j.appender.A1=org.apache.log4j.ConsoleAppender
log4j.appender.A1.layout=org.apache.log4j.PatternLayout
log4j.appender.A1.layout.ConversionPattern=%d{HH:mm:ss,SSS} %-5p [%c] - %m%n

# tighten logging on the DataNucleus Categories
log4j.category.DataNucleus.JDO=WARN, A1
log4j.category.DataNucleus.Persistence=WARN, A1
log4j.category.DataNucleus.Cache=WARN, A1
log4j.category.DataNucleus.MetaData=WARN, A1
log4j.category.DataNucleus.General=WARN, A1
log4j.category.DataNucleus.Utility=WARN, A1
log4j.category.DataNucleus.Transaction=WARN, A1
log4j.category.DataNucleus.Datastore=WARN, A1
log4j.category.DataNucleus.ClassLoading=WARN, A1
log4j.category.DataNucleus.Plugin=WARN, A1
log4j.category.DataNucleus.ValueGeneration=WARN, A1
log4j.category.DataNucleus.Enhancer=WARN, A1
log4j.category.DataNucleus.SchemaTool=WARN, A1
```

7.3 CÓDIGO FONTE DO MÓDULO GWT (PACOTE CLIENTE)

7.3.1 Interface GwtConstants.java

```

package br.ufsc.inf.tccapp.gwt.client;

public interface GwtConstants {

    String ROOT_PATH = "/";
    String flashExpressInstallPath = "js/yui/assets/expressinstall.swf";

}

```

7.3.2 Classe TccApp.java

```

package br.ufsc.inf.tccapp.gwt.client;

import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;

import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.user.client.ui.RootPanel;

/**
 * Entry point classes define <code>onModuleLoad()</code>.
 */
public class TccApp implements EntryPoint {
    /**
     * This is the entry point method.
     */
    public void onModuleLoad() {
        RootPanel.get().add(new MainPanel());
    }
}

```

7.3.3 Classe BrowsersChartDataCallback.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.content.charts.BrowsersChart;

import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;

```



```

import com.gwttext.client.widgets.form.Label;

public class BrowsersChartDataCallback<T> implements AsyncCallback<String[][]>
{
    private int height;
    private Panel panel;

    public BrowsersChartDataCallback(int height, Panel panel) {
        this.height = height;
        this.panel = panel;
    }

    /**
     * Quando há uma falha no serviço, ou na chamada do serviço este método
     * é
     * invocado.
     */
    public void onFailure(Throwable caught) {
        panel.add(new Label("Ocorreu um erro ao buscar dados do
gráfico."));
        panel.doLayout();
    }

    /**
     * Este método é invocado quando o serviço é executado com sucesso e
     * retorna
     * os dados do gráfico.
     */
    public void onSuccess(String[][] result) {
        BrowsersChart chart = new BrowsersChart(result);
        chart.setHeight(this.height);
        panel.add(chart);
        panel.doLayout();
    }
}

```

7.3.4 Classe BrowsersDashboardWidget.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.services.BrowsersChartDataService;
import br.ufsc.inf.tccapp.gwt.client.services.BrowsersChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.portal.Portlet;

public class BrowsersDashboardWidget extends Portlet {
    public BrowsersDashboardWidget() {
        setTitle("Navegadores Utilizados");
        setDraggable(false);
        setCollapsible(false);
    }
}

```

```

        /*
        * String[][] data = { { "Internet Explorer", "78" }, { "Firefox",
"18"
        * }, { "Chrome", "2" }, { "Outros", "1" } }; BrowsersChart chart
= new
        * BrowsersChart(data); chart.setHeight(170);
        * BrowsersDashboardWidget.this.add(chart);
        * BrowsersDashboardWidget.this.doLayout();
        */

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
BrowsersChartDataCallback<String[][]>(
            170, this);
        // cria o serviço
        BrowsersChartDataServiceAsync service = GWT
            .create(BrowsersChartDataService.class);
        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.5 Classe BrowsersWidget.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;
import br.ufsc.inf.tccapp.gwt.client.services.BrowsersChartDataService;
import br.ufsc.inf.tccapp.gwt.client.services.BrowsersChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;

public class BrowsersWidget extends Panel {

    private MainPanel mainPanel;

    public BrowsersWidget(MainPanel mainPanel) {
        this.mainPanel = mainPanel;

        this.setTitle("Navegadores Utilizados");
        this.setClosable(true);

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
BrowsersChartDataCallback<String[][]>(
            170, this);
        // cria o serviço
        BrowsersChartDataServiceAsync service = GWT

```

```

        .create(BrowsersChartDataService.class);
        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.6 Classe TrafficChartDataCallback.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.content.charts.TrafficChart;

import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;
import com.gwttext.client.widgets.form.Label;

public class TrafficChartDataCallback<T> implements AsyncCallback<String[][]>
{
    private int height;
    private Panel panel;

    public TrafficChartDataCallback(int height, Panel panel) {
        this.height = height;
        this.panel = panel;
    }

    /**
     * Quando há uma falha no serviço, ou na chamada do serviço este método
é
     * invocado.
     */
    public void onFailure(Throwable caught) {
        panel.add(new Label("Ocorreu um erro ao buscar dados do
gráfico."));
        panel.doLayout();
    }

    /**
     * Este método é invocado quando o serviço é executado com sucesso e
retorna
     * os dados do gráfico.
     */
    public void onSuccess(String[][] result) {
        TrafficChart chart = new TrafficChart(result);
        chart.setHeight(this.height);
        panel.add(chart);
        panel.doLayout();
    }
}

```

```
}

```

7.3.7 Classe TrafficDashboardWidget.java

```
package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.services.TrafficChartDataService;
import br.ufsc.inf.tccapp.gwt.client.services.TrafficChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.portal.Portlet;

public class TrafficDashboardWidget extends Portlet {
    public TrafficDashboardWidget() {
        setTitle("Fontes de Tráfego");
        setDraggable(false);
        setCollapsible(false);

        /*
         * String[][] data = { { "Tráfego Direto", "64", "#00b8bf" }, {
         * "Buscadores", "23", "#ffa928" }, { "Outros", "13", "#8dd5e7" }
         */
};
        * TrafficChart chart = new TrafficChart(data);
chart.setHeight(170);
        * TrafficDashboardWidget.this.add(chart);
        * TrafficDashboardWidget.this.doLayout();
        */

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
TrafficChartDataCallback<String[][]>(
            170, this);
        // cria o serviço
        TrafficChartDataServiceAsync service = GWT
            .create(TrafficChartDataService.class);
        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.8 Classe TrafficWidget.java

```
package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;
import br.ufsc.inf.tccapp.gwt.client.services.TrafficChartDataService;

```

```

import br.ufsc.inf.tccapp.gwt.client.services.TrafficChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;

public class TrafficWidget extends Panel {

    private MainPanel mainPanel;

    public TrafficWidget(MainPanel mainPanel) {
        this.mainPanel = mainPanel;

        this.setTitle("Fontes de Tráfego");
        this.setClosable(true);

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
TrafficChartDataCallback<String[][]>(
            170, this);
        // cria o serviço
        TrafficChartDataServiceAsync service = GWT
            .create(TrafficChartDataService.class);
        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.9 Classe VisitorsChartDataCallback.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.content.charts.VisitorsChart;

import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;
import com.gwttext.client.widgets.form.Label;

public class VisitorsChartDataCallback<T> implements AsyncCallback<String[][]>
{

    private int height;
    private Panel panel;

    public VisitorsChartDataCallback(int height, Panel panel) {
        this.height = height;
        this.panel = panel;
    }
}

```

```

/**
 * Quando há uma falha no serviço, ou na chamada do serviço este método
é
 * invocado.
 */
public void onFailure(Throwable caught) {
    panel.add(new Label("Ocorreu um erro ao buscar dados do
gráfico."));
    panel.doLayout();
}

/**
 * Este método é invocado quando o serviço é executado com sucesso e
retorna
 * os dados do gráfico.
 */
public void onSuccess(String[][] result) {
    VisitorsChart chart = new VisitorsChart(result);
    chart.setHeight(this.height);
    panel.add(chart);
    panel.doLayout();
}
}

```

7.3.10 Classe VisitorsDashboardWidget.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.services.VisitorsChartDataService;
import br.ufsc.inf.tccapp.gwt.client.services.VisitorsChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.portal.Portlet;

public class VisitorsDashboardWidget extends Portlet {

    /**
     * Construtor do widget
     */
    public VisitorsDashboardWidget() {
        // seta alguns atributos do gráfico
        setTitle("Visitas");
        setDraggable(false);
        setCollapsible(false);

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
VisitorsChartDataCallback<String[][]>(
            170, this);

        // cria o serviço
        VisitorsChartDataServiceAsync service = GWT
            .create(VisitorsChartDataService.class);
    }
}

```

```

        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.11 Classe VisitorsWidget.java

```

package br.ufsc.inf.tccapp.gwt.client.content;

import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;
import br.ufsc.inf.tccapp.gwt.client.services.VisitorsChartDataService;
import br.ufsc.inf.tccapp.gwt.client.services.VisitorsChartDataServiceAsync;

import com.google.gwt.core.client.GWT;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.gwttext.client.widgets.Panel;

public class VisitorsWidget extends Panel {

    private MainPanel mainPanel;

    public VisitorsWidget(MainPanel mainPanel) {
        this.mainPanel = mainPanel;

        this.setTitle("Visitas");
        this.setClosable(true);

        // cria o objeto que irá tratar o retorno do serviço
        AsyncCallback<String[][]> callback = new
VisitorsChartDataCallback<String[][]>(
            170, this);
        // cria o serviço
        VisitorsChartDataServiceAsync service = GWT
            .create(VisitorsChartDataService.class);
        // executa o serviço, passando como parâmetro o objeto que tratará
seu
        // retorno
        service.getData(callback);
    }
}

```

7.3.12 Classe BrowsersChart.java

```

package br.ufsc.inf.tccapp.gwt.client.content.charts;

import com.gwtext.client.data.ArrayReader;
import com.gwtext.client.data.FieldDef;
import com.gwtext.client.data.IntegerFieldDef;
import com.gwtext.client.data.MemoryProxy;
import com.gwtext.client.data.RecordDef;
import com.gwtext.client.data.Store;
import com.gwtext.client.data.StringFieldDef;
import com.gwtext.client.widgets.chart.yui.ColumnChart;

public class BrowsersChart extends ColumnChart {
    public BrowsersChart(Object[][] data) {
        setExpressInstall("js/yui/assets/expressinstall.swf");
        setWMode("transparent");

        setXField("browsers");
        setYField("count");
        setStore(createStore(data));
    }

    private Store createStore(Object[][] data) {
        // cria o proxy dos dados
        MemoryProxy proxy = new MemoryProxy(data);
        // cria os conjuntos de dados
        RecordDef recordDef = new RecordDef(new FieldDef[] {
            new StringFieldDef("browsers"), new
IntegerFieldDef("count") });
        // cria o leitor de dados
        ArrayReader reader = new ArrayReader(recordDef);
        // cria o store
        Store store = new Store(proxy, reader);
        store.load();
        return store;
    }
}

```

7.3.13 Classe TrafficChart.java

```

package br.ufsc.inf.tccapp.gwt.client.content.charts;

import com.gwtext.client.data.ArrayReader;
import com.gwtext.client.data.FieldDef;
import com.gwtext.client.data.IntegerFieldDef;
import com.gwtext.client.data.MemoryProxy;
import com.gwtext.client.data.RecordDef;
import com.gwtext.client.data.Store;
import com.gwtext.client.data.StringFieldDef;
import com.gwtext.client.widgets.chart.yui.PieChart;

public class TrafficChart extends PieChart {
    public TrafficChart(Object[][] data) {

```



```

        setExpressInstall("js/yui/assets/expressinstall.swf");
        setWMode("transparent");

        setDataField("count");
        setCategoryField("response");
        setStore(createStore(data));
    }

    private Store createStore(Object[][] data) {
        // cria o proxy dos dados
        MemoryProxy proxy = new MemoryProxy(data);
        // cria os conjuntos de dados
        RecordDef recordDef = new RecordDef(new FieldDef[] {
            new StringFieldDef("response"), new
IntegerFieldDef("count"),
            new StringFieldDef("legend") });
        // cria o leitor de dados
        ArrayReader reader = new ArrayReader(recordDef);
        // cria o store
        Store store = new Store(proxy, reader);
        store.load();
        return store;
    }
}

```

7.3.14 Classe VisitorsChart.java

```

package br.ufsc.inf.tccapp.gwt.client.content.charts;

import br.ufsc.inf.tccapp.gwt.client.GwtConstants;
import com.gwttext.client.data.ArrayReader;
import com.gwttext.client.data.FieldDef;
import com.gwttext.client.data.FloatFieldDef;
import com.gwttext.client.data.MemoryProxy;
import com.gwttext.client.data.RecordDef;
import com.gwttext.client.data.Store;
import com.gwttext.client.data.StringFieldDef;
import com.gwttext.client.widgets.chart.yui.LineChart;
import com.gwttext.client.widgets.chart.yui.NumericAxis;

public class VisitorsChart extends LineChart {
    public VisitorsChart(Object[][] data) {
        store = createStore(data);
        store.load();
        setExpressInstall(GwtConstants.flashExpressInstallPath);
        setWMode("transparent");
        setStore(store);
        setYField("visitantes");
        setXField("dia");
        setBorder(true);
    }
}

```

```

public void setInitialYValue(Integer value) {
    NumericAxis yAxis = new NumericAxis();
    yAxis.setMinimum(value);
    setYAxis(yAxis);
}

private Store createStore(Object[][] data) {
    // seta os dados
    MemoryProxy proxy = new MemoryProxy(data);
    RecordDef recordDef = new RecordDef(new FieldDef[] {
        new StringFieldDef("dia"), new
FloatFieldDef("visitantes") });
    ArrayReader reader = new ArrayReader(recordDef);
    return new Store(proxy, reader);
}

public Store getStore() {
    return this.store;
}
}

```

7.3.15 Classe Dashboard.java

```

package br.ufsc.inf.tccapp.gwt.client.content.dashboard;

import br.ufsc.inf.tccapp.gwt.client.content.BrowsersDashboardWidget;
import br.ufsc.inf.tccapp.gwt.client.content.TrafficDashboardWidget;
import br.ufsc.inf.tccapp.gwt.client.content.VisitorsDashboardWidget;
import br.ufsc.inf.tccapp.gwt.client.mainpanel.MainPanel;

import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.ui.Button;
import com.gwttext.client.widgets.Panel;
import com.gwttext.client.widgets.form.Label;
import com.gwttext.client.widgets.layout.ColumnLayoutData;
import com.gwttext.client.widgets.layout.FitLayout;
import com.gwttext.client.widgets.portal.Portal;
import com.gwttext.client.widgets.portal.PortalColumn;
import com.gwttext.client.widgets.portal.Portlet;

public class Dashboard {
    private MainPanel mainPanel;
    private Panel panel = createPanel();

    public Dashboard(MainPanel mainPanel) {
        this.mainPanel = mainPanel;
    }

    public Panel getPanel() {
        return panel;
    }
}

```

```

private Panel createPanel() {
    Panel panel = new Panel();
    panel.setTitle("Dashboard");
    panel.setBorder(false);
    panel.setPaddings(15);
    panel.setLayout(new FitLayout());

    // create a portal
    Portal portal = new Portal();
    portal.setBorder(false);

    // create portal columns
    PortalColumn uniqueCol = new PortalColumn();
    uniqueCol.setPaddings(10, 10, 0, 10);

    // create and add portlets
    addPortlets(uniqueCol);

    // add column portal to portal
    portal.add(uniqueCol, new ColumnLayoutData(.99));
    // add portal to panel
    panel.add(portal);

    return panel;
}

/**
 * Adiciona portlets ao portal
 */
private void addPortlets(PortalColumn uniqueCol) {
    //addVisitorsPortlet(uniqueCol);
    uniqueCol.add(new VisitorsDashboardWidget());
    uniqueCol.add(new BrowsersDashboardWidget());
    uniqueCol.add(new TrafficDashboardWidget());
}

private void addVisitorsPortlet(PortalColumn uniqueCol) {
    final Portlet p = new Portlet();
    p.setTitle("Teste");

    Button b = new Button();
    b.setText("botao a");
    b.addClickHandler(new ClickHandler() {
        public void onClick(ClickEvent event) {
            p.add(new Label("yes!"));
            p.doLayout();
        }
    });

    p.add(b);
    uniqueCol.add(p);
}
}

```



```

                                boolean isNotSelf =
!component.getId().equals(
    CenterPanel.this.getActiveTab().getId());
                                boolean isCloseable = true;
                                try {
component).isClosable();
                                    isCloseable = ((Panel)
                                } catch (Exception classCastEx) {
                                }
                                if (isNotSelf && isCloseable) {
                                    CenterPanel.this.remove(component);
                                }
                            }
                    }
                });
                menu.addItem(closeOthers);
            }

            BaseItem close = menu.getItem("close-tab-item");
            if (this.getActiveTab().isClosable()) {
                close.enable();
            } else {
                close.disable();
            }
            BaseItem closeOthers = menu.getItem("close-others-item");
            if (this.getItems().length == 1) {
                closeOthers.disable();
            } else {
                closeOthers.enable();
            }
            menu.showAt(e.getXY());
        }
    }
}

```

7.3.17 Classe MainPanel.java

```

package br.ufsc.inf.tccapp.gwt.client.mainpanel;

import com.google.gwt.user.client.ui.Widget;
import com.gwttext.client.core.Margins;
import com.gwttext.client.core.RegionPosition;
import com.gwttext.client.widgets.Panel;
import com.gwttext.client.widgets.Viewport;
import com.gwttext.client.widgets.layout.BorderLayout;
import com.gwttext.client.widgets.layout.BorderLayoutData;
import com.gwttext.client.widgets.layout.FitLayout;

public class MainPanel extends Panel {

    private Widget northPanel = new TopPanel();

```

```

private Panel westPanel = new MenuPanel(this);
private Panel centerPanel = new CenterPanel(this);

public MainPanel() {
    this.setPadding(10);
    this.setBorder(false);
    this.setLayout(new FitLayout());

    // northPanel - topo
    Panel borderPanel = new Panel();
    borderPanel.setLayout(new BorderLayout());
    borderPanel.add(northPanel, new
BorderLayoutData(RegionPosition.NORTH));

    // centerPanel - conteudo
    BorderLayoutData centerData = new BorderLayoutData(
        RegionPosition.CENTER);
    centerData.setMargins(3, 3, 3, 3);
    borderPanel.add(centerPanel, centerData);

    // westPanel - menu
    BorderLayoutData westData = new
BorderLayoutData(RegionPosition.WEST);
    westData.setMargins(3, 3, 3, 3);
    westData.setCMargins(new Margins(3, 3, 3, 3));
    borderPanel.add(westPanel, westData);

    this.add(borderPanel);
    new Viewport(this);
}

public void setContent(Widget content) {
    this.centerPanel.add(content);
}
}

```

7.3.18 Classe MenuPanel.java

```

package br.ufsc.inf.tccapp.gwt.client.mainpanel;

import br.ufsc.inf.tccapp.gwt.client.content.BrowsersWidget;
import br.ufsc.inf.tccapp.gwt.client.content.TrafficWidget;
import br.ufsc.inf.tccapp.gwt.client.content.VisitorsWidget;

import com.google.gwt.user.client.ui.Widget;
import com.gwttext.client.core.EventObject;
import com.gwttext.client.data.Node;
import com.gwttext.client.widgets.Panel;
import com.gwttext.client.widgets.tree.TreeNode;
import com.gwttext.client.widgets.tree.TreePanel;
import com.gwttext.client.widgets.tree.event.TreeNodeListenerAdapter;

```

```

public class MenuPanel extends Panel {
    private MainPanel mainPanel;

    public MenuPanel(MainPanel mainPanel) {
        this.mainPanel = mainPanel;

        setTitle("Menu");
        setCollapsible(true);
        setWidth(150);
        setPadding(10);

        add(new MenuTree());
    }

    private class MenuTree extends TreePanel {
        public MenuTree() {
            setBorder(false);

            TreeNode root = new TreeNode("Gráficos");
            root.setExpanded(true);

            TreeNode childVisitantes = new TreeNode();
            childVisitantes.setLeaf(true);
            childVisitantes.setText("Visitantes");
            childVisitantes.addListener(new TccAppMenuTreeNodeListener(
                new VisitorsWidget(mainPanel)));

            TreeNode childBrowsers = new TreeNode();
            childBrowsers.setLeaf(true);
            childBrowsers.setText("Navegadores Utilizados");
            childBrowsers.addListener(new TccAppMenuTreeNodeListener(
                new BrowsersWidget(mainPanel)));

            TreeNode childTrafego = new TreeNode();
            childTrafego.setLeaf(true);
            childTrafego.setText("Fontes de Tráfego");
            childTrafego.addListener(new TccAppMenuTreeNodeListener(
                new TrafficWidget(mainPanel)));

            root.appendChild(childVisitantes);
            root.appendChild(childBrowsers);
            root.appendChild(childTrafego);

            setRootNode(root);
        }
    }

    private class TccAppMenuTreeNodeListener extends TreeNodeListenerAdapter
    {
        private Widget content;

        public TccAppMenuTreeNodeListener(Widget content) {
            this.content = content;
        }

        @Override
        public void onClick(Node node, EventObject e) {
            mainPanel.setContent(content);
        }
    }
}

```

```

    }
}

```

7.3.19 Classe TopPanel.java

```

package br.ufsc.inf.tccapp.gwt.client.mainpanel;

import br.ufsc.inf.tccapp.gwt.client.GwtConstants;

import com.google.gwt.user.client.ui.HorizontalPanel;

public class TopPanel extends HorizontalPanel {

    public TopPanel() {
        this.setVerticalAlignment(HorizontalPanel.ALIGN_BOTTOM);
        this.setSpacing(10);

        this.add(new
com.google.gwt.user.client.ui.Image(GwtConstants.ROOT_PATH
+ "images/logo.png"));
    }
}

```

7.3.20 Interface BrowsersChartDataService.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.RemoteService;
import com.google.gwt.user.client.rpc.RemoteServiceRelativePath;

@RemoteServiceRelativePath("browserschartdataservice")
public interface BrowsersChartDataService extends RemoteService {
    String[][] getData();
}

```

7.3.21 Interface BrowsersChartDataServiceAsync.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.AsyncCallback;

```



```

public interface BrowsersChartDataServiceAsync {
    void getData(AsyncCallback<String[][]> callback);
}

```

7.3.22 Interface TrafficChartDataService.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.RemoteService;
import com.google.gwt.user.client.rpc.RemoteServiceRelativePath;

@RemoteServiceRelativePath("trafficchartdataservice")
public interface TrafficChartDataService extends RemoteService {
    String[][] getData();
}

```

7.3.23 Interface TrafficChartDataServiceAsync.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.AsyncCallback;

public interface TrafficChartDataServiceAsync {
    void getData(AsyncCallback<String[][]> callback);
}

```

7.3.24 Interface VisitorsChartDataService.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.RemoteService;
import com.google.gwt.user.client.rpc.RemoteServiceRelativePath;

@RemoteServiceRelativePath("visitorschartdataservice")
public interface VisitorsChartDataService extends RemoteService {
    String[][] getData();
}

```

7.3.25 Interface VisitorsChartDataServiceAsync.java

```

package br.ufsc.inf.tccapp.gwt.client.services;

import com.google.gwt.user.client.rpc.AsyncCallback;

public interface VisitorsChartDataServiceAsync {

    void getData(AsyncCallback<String[][]> callback);

}

```

7.4 CÓDIGO FONTE DO MÓDULO GWT (PACOTE SERVIDOR)

7.4.1 Classe BrowsersChartDataServiceImpl.java

```

package br.ufsc.inf.tccapp.gwt.server.services;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

import br.ufsc.inf.tccapp.ejb.data.browserschartdata.BrowsersChartDataEjb;
import br.ufsc.inf.tccapp.ejb.data.browserschartdata.BrowsersChartDataEjbRemote;
import br.ufsc.inf.tccapp.gwt.client.services.BrowsersChartDataService;

import com.google.gwt.user.server.rpc.RemoteServiceServlet;

/**
 * @author Tiago Braun
 *
 */
@SuppressWarnings("serial")
public class BrowsersChartDataServiceImpl extends RemoteServiceServlet
    implements BrowsersChartDataService {

    public String[][] getData() {

        Context ctx;
        BrowsersChartDataEjbRemote ejb = null;
        try {
            ctx = new InitialContext();
            ejb = (BrowsersChartDataEjbRemote) ctx
                .lookup(BrowsersChartDataEjb.RemoteJNDIName);
        } catch (NamingException e) {
            // TODO LOG
        }
    }
}

```

```

        String[][] data =.ejb.getData();

        return data;
    }
}

```

7.4.2 Classe TrafficChartDataServiceImpl.java

```

package br.ufsc.inf.tccapp.gwt.server.services;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

import br.ufsc.inf.tccapp.ejb.data.trafficchartdata.TrafficChartDataEjb;
import br.ufsc.inf.tccapp.ejb.data.trafficchartdata.TrafficChartDataEjbRemote;
import br.ufsc.inf.tccapp.gwt.client.services.TrafficChartDataService;

import com.google.gwt.user.server.rpc.RemoteServiceServlet;

/**
 * @author Tiago Braun
 *
 */
@SuppressWarnings("serial")
public class TrafficChartDataServiceImpl extends RemoteServiceServlet
implements TrafficChartDataService {

    public String[][] getData() {

        Context ctx;
        TrafficChartDataEjbRemote.ejb = null;
        try {
            ctx = new InitialContext();
           .ejb = (TrafficChartDataEjbRemote) ctx
                .lookup(TrafficChartDataEjb.RemoteJNDIName);
        } catch (NamingException e) {
            // TODO LOG
        }

        String[][] data = .ejb.getData();

        return data;
    }
}

```

7.4.3 Classe VisitorsChartDataServiceImpl.java

```

package br.ufsc.inf.tccapp.gwt.server.services;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

import br.ufsc.inf.tccapp.ejb.data.visitorschartdata.VisitorsChartDataEjb;
import br.ufsc.inf.tccapp.ejb.data.visitorschartdata.VisitorsChartDataEjbRemote;
import br.ufsc.inf.tccapp.gwt.client.services.VisitorsChartDataService;

import com.google.gwt.user.server.rpc.RemoteServiceServlet;

/**
 * @author Tiago Braun
 *
 */
@SuppressWarnings("serial")
public class VisitorsChartDataServiceImpl extends RemoteServiceServlet
    implements VisitorsChartDataService {

    public String[][] getData() {

        Context ctx;
        VisitorsChartDataEjbRemote ejb = null;
        try {
            ctx = new InitialContext();
            ejb = (VisitorsChartDataEjbRemote) ctx
                .lookup(VisitorsChartDataEjb.RemoteJNDIName);
        } catch (NamingException e) {
            // TODO LOG
        }

        String[][] data = ejb.getData();

        return data;
    }
}

```

7.5 CÓDIGO FONTE DO MÓDULO EJB

7.5.1 Classe BrowsersChartDataEjb.java

```

package br.ufsc.inf.tccapp.ejb.data.browserschartdata;

```

```

import javax.ejb.Stateless;
import javax.xml.xpath.XPathConstants;

import org.jboss.ejb3.annotation.LocalBinding;
import org.jboss.ejb3.annotation.RemoteBinding;
import org.jboss.ejb3.annotation.RemoteBindings;

import br.ufsc.inf.tccapp.TccAppConstants;
import br.ufsc.inf.tccapp.helper.XPathHelper;

/**
 *
 * Classe busca os dados no xml e entrega os dados prontos para o gráfico.
 *
 * @author Tiago Braun
 *
 */
@Stateless
@LocalBinding(jndiBinding = "local/BrowsersChartDataEjb")
@RemoteBindings( { @RemoteBinding(jndiBinding = "remote/BrowsersChartDataEjb")
})
public class BrowsersChartDataEjb implements BrowsersChartDataEjbLocal,
        BrowsersChartDataEjbRemote {

    public static final String RemoteJNDIName =
"remote/BrowsersChartDataEjb";
    public static final String LocalJNDIName = "local/BrowsersChartDataEjb";

    private String queryInternetExplorer =
"sum(//dia/navegadores/navegador[normalize-space(nome)='internet
explorer']/acessos)";
    private String queryFirefox =
"sum(//dia/navegadores/navegador[normalize-space(nome)='firefox']/acessos)";
    private String queryChrome = "sum(//dia/navegadores/navegador[normalize-
space(nome)='chrome']/acessos)";
    private String queryOutros = "sum(//dia/navegadores/navegador[normalize-
space(nome)='outros']/acessos)";

    private Double totalInternetExplorer, totalFirefox, totalChrome,
        totalOutros;

    public String[][] getData() {
        retrieveXmlData();
        return new String[][] {
            { "Internet Explorer",
totalInternetExplorer.toString() },
            { "Mozilla Firefox", totalFirefox.toString() },
            { "Google Chrome", totalChrome.toString() },
            { "Outros", totalOutros.toString() } };
    }

    private void retrieveXmlData() {
        XPathHelper xPath = new XPathHelper(
TccAppConstants.XML_DATA_FILE_NAVEGADORES_FONTESTRAFEGO);
        try {
            totalInternetExplorer = (Double) xPath.processXPath(
                queryInternetExplorer, XPathConstants.NUMBER);

```

```

        totalFirefox = (Double) XPath.processXPath(queryFirefox,
            XPathConstants.NUMBER);
        totalChrome = (Double) XPath.processXPath(queryChrome,
            XPathConstants.NUMBER);
        totalOutros = (Double) XPath.processXPath(queryOutros,
            XPathConstants.NUMBER);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) {
    String[][] data = new BrowsersChartDataEjb().getData();
    for (String[] array : data) {
        for (String s : array) {
            System.out.print(s);
        }
        System.out.println();
    }
}
}
}

```

7.5.2 Interface BrowsersChartDataEjbLocal.java

```

package br.ufsc.inf.tccapp.ejb.data.browserschartdata;

import javax.ejb.Local;

@Local
public interface BrowsersChartDataEjbLocal {
    String[][] getData();
}

```

7.5.3 Interface BrowsersChartDataEjbRemote.java

```

package br.ufsc.inf.tccapp.ejb.data.browserschartdata;

import javax.ejb.Remote;

@Remote
public interface BrowsersChartDataEjbRemote {
    String[][] getData();
}

```

7.5.4 Classe TrafficChartDataEjb.java

```

package br.ufsc.inf.tccapp.ejb.data.trafficchartdata;

import javax.ejb.Stateless;
import javax.xml.xpath.XPathConstants;

import org.jboss.ejb3.annotation.LocalBinding;
import org.jboss.ejb3.annotation.RemoteBinding;
import org.jboss.ejb3.annotation.RemoteBindings;

import br.ufsc.inf.tccapp.TccAppConstants;
import br.ufsc.inf.tccapp.helper.XPathHelper;

/**
 *
 * Classe busca os dados no xml e retorna no formato necessário para o
 gráfico.
 *
 * @author Tiago Braun
 *
 */
@Stateless
@LocalBinding(jndiBinding = "local/TrafficChartDataEjb")
@RemoteBindings( { @RemoteBinding(jndiBinding = "remote/TrafficChartDataEjb")
})
public class TrafficChartDataEjb implements TrafficChartDataEjbLocal,
    TrafficChartDataEjbRemote {

    public static final String RemoteJNDIName =
"remote/TrafficChartDataEjb";
    public static final String LocalJNDIName = "local/TrafficChartDataEjb";

    private static String querySomaTotal =
"sum(//dia/fontestrafego/fontetrafeago/acessos)";
    private static String querySomaDireto =
"sum(//dia/fontestrafego/fontetrafeago[normalize-
space(tipo)='direto']/acessos)";
    private static String querySomaPesquisa =
"sum(//dia/fontestrafego/fontetrafeago[normalize-
space(tipo)='pesquisa']/acessos)";
    private static String querySomaReferencia =
"sum(//dia/fontestrafego/fontetrafeago[normalize-
space(tipo)='referencia']/acessos)";

    private Double somaTotal, somaDireto, somaPesquisa, somaReferencia;

    private Double prcDireto, prcPesquisa, prcReferencia;

    public String[][] getData() {
        retrieveXmlData();
        processXmlData();
        return new String[][] { { "Direto", prcDireto.toString(),
"#00b8bf" },
                                { "Pesquisa", prcPesquisa.toString(), "#ffa928" },

```

```

        { "Referência", prcReferencia.toString(), "#8dd5e7" }
};

}

/**
 * Recupera os dados de quantidade de acessos no xml.
 */
private void retrieveXmlData() {
    XPathHelper xPath = new XPathHelper(
TccAppConstants.XML_DATA_FILE_NAVEGADORES_FONTESTRAFEGO);
    try {
        somaTotal = (Double) xPath.processXPath(querySomaTotal,
XPathConstants.NUMBER);
        somaDireto = (Double) xPath.processXPath(querySomaDireto,
XPathConstants.NUMBER);
        somaPesquisa = (Double)
xPath.processXPath(querySomaPesquisa,
XPathConstants.NUMBER);
        somaReferencia = (Double)
xPath.processXPath(querySomaReferencia,
XPathConstants.NUMBER);
    } catch (Exception e) {
        // TODO LOGAR
    }
}

/**
 * Processa os dados recuperados no xml, calculando as porcentagens.
 */
private void processXmlData() {
    prcDireto = getPercentage(somaDireto);
    prcPesquisa = getPercentage(somaPesquisa);
    prcReferencia = getPercentage(somaReferencia);
}

/**
 * Retorna o valor em porcentagem da fração dada.
 *
 * @param valueFraction
 * @return
 */
private Double getPercentage(Double valueFraction) {
    return valueFraction * 100 / somaTotal;
}
}

```

7.5.5 Interface TrafficChartDataEjbLocal.java

```

package br.ufsc.inf.tccapp.ejb.data.trafficchartdata;

```



```

import javax.ejb.Local;

@Local
public interface TrafficChartDataEjbLocal {
    String[][] getData();
}

```

7.5.6 Interface TrafficChartDataEjbRemote.java

```

package br.ufsc.inf.tccapp.ejb.data.trafficchartdata;

import javax.ejb.Remote;

@Remote
public interface TrafficChartDataEjbRemote {
    String[][] getData();
}

```

7.5.7 Classe VisitorsChartDataEjb.java

```

package br.ufsc.inf.tccapp.ejb.data.visitorschartdata;

import java.util.ArrayList;

import javax.ejb.Stateless;
import javax.xml.xpath.XPathConstants;

import org.jboss.ejb3.annotation.LocalBinding;
import org.jboss.ejb3.annotation.RemoteBinding;
import org.jboss.ejb3.annotation.RemoteBindings;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import br.ufsc.inf.tccapp.TccAppConstants;
import br.ufsc.inf.tccapp.helper.XPathHelper;

@Stateless
@LocalBinding(jndiBinding = "local/VisitorsChartDataEjb")
@RemoteBindings( { @RemoteBinding(jndiBinding = "remote/VisitorsChartDataEjb")
})
public class VisitorsChartDataEjb implements VisitorsChartDataEjbLocal,
    VisitorsChartDataEjbRemote {

    public static final String RemoteJNDIName =
"remote/VisitorsChartDataEjb";
    public static final String LocalJNDIName = "local/VisitorsChartDataEjb";

    // private String query = "//dia/data | //dia/acessos";

```

```

    private String query =
        "//acessos/ acesso/meses/mes/semanas/semana/datas/data/nome |
        //acessos/ acesso/meses/mes/semanas/semana/datas/data/qtde";

    public String[][] getData() {
        NodeList nodeList = getNodeList();
        return processNodeList(nodeList);
    }

    private NodeList getNodeList() {
        XPathHelper xPath = new XPathHelper(
            TccAppConstants.XML_DATA_FILE_ACESSOS);
        NodeList nodeList = null;
        try {
            nodeList = (NodeList) xPath.processXPath(query,
                XPathConstants.NODESET);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return nodeList;
    }

    private String[][] processNodeList(NodeList nodeList) {
        ArrayList<String[]> list = new ArrayList<String[]>();
        ArrayList<String> row = new ArrayList<String>();
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            String nodeName = node.getNodeName();
            if (nodeName.equals("nome")) {
                row.add(node.getTextContent());
            } else if (nodeName.equals("qtde")) {
                row.add(node.getTextContent());
                list.add(row.toArray(new String[] {}));
                row.clear();
            }
        }
        return list.toArray(new String[][] {});
    }

    public static void main(String[] args) {
        Object[][] data = new VisitorsChartDataEjb().getData();
        for (Object[] row : data) {
            for (Object item : row) {
                System.out.print(item + "\t");
            }
            System.out.println();
        }
    }
}

```

7.5.8 Interface VisitorsChartDataEjbLocal.java

```

package br.ufsc.inf.tccapp.ejb.data.visitorschartdata;

import javax.ejb.Local;

@Local
public interface VisitorsChartDataEjbLocal {
    String[][] getData();
}

```

7.5.9 Interface VisitorsChartDataEjbRemote.java

```

package br.ufsc.inf.tccapp.ejb.data.visitorschartdata;

import javax.ejb.Remote;

@Remote
public interface VisitorsChartDataEjbRemote {
    String[][] getData();
}

```

7.6 CÓDIGO FONTE DO MÓDULO DE UTILITÁRIOS TCCAPP.JAR

7.6.1 Interface TccAppConstants.java

```

package br.ufsc.inf.tccapp;

import java.io.File;

public interface TccAppConstants {

    /**
     * Path para a pasta de arquivos da aplicação.
     * D:\braun\ufsc\tcc\programa\deploy\tccapp.ear\data
     */
    String DATA_PATH = "D:" + File.separator + "braun" + File.separator
        + "ufsc" + File.separator + "tcc" + File.separator +
"programa"
        + File.separator + "deploy" + File.separator + "tccapp.ear"
        + File.separator + "data";

    File XML_DATA_FILE_NAVEGADORES_FONTESTRAFEGO = new File(DATA_PATH
        + File.separator + "dadosnavegadoresefontestrafego.xml");

    File XML_DATA_FILE_ACESSOS = new File(DATA_PATH + File.separator
        + "dadosacessos.xml");
}

```

```
String flashExpressInstallPath = "js/yui/assets/expressinstall.swf";
}
```

7.6.2 Classe XPathHelper.java

```
package br.ufsc.inf.tccapp.helper;

import java.io.File;
import java.io.IOException;

import javax.xml.namespace.QName;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;

import org.w3c.dom.Document;
import org.xml.sax.SAXException;

import com.sun.org.apache.xerces.internal.jaxp.DocumentBuilderFactoryImpl;

public class XPathHelper {

    private File xml;

    public XPathHelper(File xml) {
        this.xml = xml;
    }

    /**
     * Executa a query no xml e retorna um NodeList da consulta.
     *
     * @param query
     * @return
     * @throws IOException
     * @throws ParserConfigurationException
     * @throws SAXException
     * @throws XPathExpressionException
     */
    public Object processXPath(String query, QName returnType)
        throws IOException, ParserConfigurationException,
        SAXException,
        XPathExpressionException {
        DocumentBuilderFactory factory = DocumentBuilderFactoryImpl
            .newInstance();
        factory.setValidating(false);
        factory.setIgnoringComments(true);
        factory.setIgnoringElementContentWhitespace(true);
        DocumentBuilder docB;
```

```

        docB = factory.newDocumentBuilder();
        Document doc = docB.parse(xml);
        XPathFactory xF = XPathFactory.newInstance();
        XPath xp = xF.newXPath();
        return xp.evaluate(query, doc, returnType);
    }
}

```

7.7 CÓDIGO FONTE DO MÓDULO FLEX (SERVIDOR)

7.7.1 Classe TCC_APPServlet.java

```

package br.ufsc.inf.tccapp.flex;

import java.io.IOException;
import javax.servlet.http.*;

import br.ufsc.inf.tccapp.flex.xml.XmlVisitantes;

@SuppressWarnings("serial")
public class TCC_APPServlet extends HttpServlet {

    XmlVisitantes visitantes = new XmlVisitantes();

    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println(getVisitantes());
    }

    public TCC_APPServlet(){

    }

    public String getVisitantes() {
        return visitantes.getXml();
    }
}

```

7.7.2 Classe XmlVisitantes.java

```

package br.ufsc.inf.tccapp.flex.xml;

```

```

import java.io.BufferedReader;
import java.io.FileReader;

import br.ufsc.inf.tccapp.TccAppConstants;

public class XmlVisitantes {

    public String getXml() {
        StringBuilder sb = new StringBuilder();
        FileReader fr;
        try {
            fr = new FileReader(TccAppConstants.XML_DATA_FILE_ACESSOS);
            BufferedReader br = new BufferedReader(fr);
            String line;
            while ((line = br.readLine()) != null) {
                sb.append(line);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return sb.toString();
    }
}

```

7.8 CÓDIGO FONTE DO MÓDULO FLEX (CLIENTE)

7.8.1 Interface de inicialização TCC_Application.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical"
    xmlns:comp="Components.*" >
    <mx:Label fontSize="15" color="white" fontWeight="bold" text="TCC
Application"/>
    <mx:Accordion height="500" width="800" id="abas">
        <mx:Canvas label="DashBoard" width="100%" height="100%">
            <mx:VBox >
                <comp:DashBoardVisitante_Component
id="dashBoardVisitante" />
                <mx:HBox>
                    <comp:DashBoardNavegador_Component
id="dashBoardNavegador" />
                    <comp:DashBoardTrafego_Component
id="dashBoardTrafego" />
                </mx:HBox>
            </mx:VBox>
        </mx:Canvas>
        <mx:Canvas label="Visitantes" width="100%" height="100%">

```

```

        <comp:Visitante_Component id="visitante" />
    </mx:Canvas>
    <mx:Canvas label="Navegadores" width="100%" height="100%">
        <comp:Navegadores_Components id="navegador" />
    </mx:Canvas>
    <mx:Canvas label="Fonte de Tráfego" width="100%" height="100%">
        <comp:Trafego_Component id="trafego" />
    </mx:Canvas>
</mx:Accordion>
</mx:Application>

```

7.8.2 Componente Flex Visitante_Componet.xml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical"
width="796" height="406"
    creationComplete="initApp();" >
    <mx:SeriesInterpolate duration="600" id="showData" />
    <mx:VBox width="750" height="25">
        <mx:HBox width="750" horizontalAlign="right" >
            <mx:Label text="Gráfico por:" x="30" y="60" />
            <mx:Button label="Ano" x="60" y="60"
click="atualizaPeriodoAno()" />
        </mx:HBox>
    </mx:VBox>
    <mx:VBox width="750" height="300">
        <mx:LineChart dataProvider="{dadosInputPeriodos}"
showDataTips="true" itemClick="updateDetails(event)" width="750"
height="270">
            <mx:horizontalAxis>
                <mx:CategoryAxis categoryField="nome"
dataProvider="{dadosInputPeriodos}" />
            </mx:horizontalAxis>
            <mx:series>
                <mx:LineSeries yField="qtde" displayName="Quantidade"
showDataEffect="showData" />
            </mx:series>
        </mx:LineChart>
    </mx:VBox>
    <mx:HTTPService
        id="service"
        url="http://localhost:8080/jsp/requisicao.jsp"
        result="resultHandler(event)"
        fault="faultHandler(event)"
        method="GET"/>
    <mx:Script source="visitante.as" />
</mx:TitleWindow>

```

7.8.3 Componente Flex Navegadores_Components.xml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical"
width="796" height="406">
  <mx:VBox>
    <mx:BarChart height="325" width="750"
paddingLeft="5" paddingRight="5" dataProvider="{navegadores}"
showDataTips="true">
      <mx:verticalAxis>
        <mx:CategoryAxis categoryField="nome" />
      </mx:verticalAxis>
      <mx:series>
        <mx:BarSeries
xField="porcentagem"
        />
      </mx:series>
    </mx:BarChart>
  </mx:VBox>
  <mx:Script source="navegador.as" />
</mx:TitleWindow>

```

7.8.4 Componente Flex Trafego_Component.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical"
width="796" height="406">
  <mx:Box>
    <mx:PieChart height="250" dataProvider="{fonteTrafego}"
width="750" id="myPieChart" >
      <mx:series>
        <mx:PieSeries field="Quantidade"
labelPosition="callout" nameField="fonteTipo" labelFunction="chartLabel">
          <mx:fills>
            <mx:RadialGradient>
              <mx:entries>
                <mx:GradientEntry
color="#E9C836" />
                <mx:GradientEntry
color="#AA9127" />
              </mx:entries>
            </mx:RadialGradient>
            <mx:RadialGradient>
              <mx:entries>
                <mx:GradientEntry
color="#A1AECF" />
                <mx:GradientEntry
color="#47447A" />
              </mx:entries>
            </mx:RadialGradient>
            <mx:RadialGradient>
              <mx:entries>
                <mx:GradientEntry
color="#339933" />
                <mx:GradientEntry

```



```

color="#339998" />
                                </mx:entries>
                                </mx:RadialGradient>
                                </mx:fills>
                                </mx:PieSeries>
                                </mx:series>
                                </mx:PieChart>
                                </mx:Box>
                                <mx:VBox horizontalAlign="center" width="750" height="70"
borderStyle="outset" >
                                <mx:Label text="Legenda" fontSize="12" fontStyle="normal"
textAlign="center" fontWeight="bold" width="730" y="0"/>
                                <mx:Legend dataProvider="{myPieChart}" textAlign="center"
height="31" width="732" y="28"/>
                                </mx:VBox>
                                <mx:Script source="trafego.as" />
</mx:TitleWindow>

```

7.8.5 Componente Flex DashBoardNavegador_Component.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow title="Navegadores Utilizados"
xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical" width="390"
height="240">
    <mx:VBox>
        <mx:BarChart height="187" width="353"
paddingLeft="5" paddingRight="5" dataProvider="{navegadores}"
showDataTips="true">
            <mx:verticalAxis>
                <mx:CategoryAxis categoryField="nome"/>
            </mx:verticalAxis>
            <mx:series>
                <mx:BarSeries
                    xField="porcentagem"
                />
            </mx:series>
        </mx:BarChart>
    </mx:VBox>
    <mx:Script source="navegador.as" />
</mx:TitleWindow>

```

7.8.6 Componente Flex DashBoardTrafego_Component.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow title="Fontes de tráfego"
xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical" width="390"
height="240">
    <mx:Box>
        <mx:PieChart verticalCenter="true" height="189"

```

```

dataProvider="{fonteTrafego}" width="359" id="myPieChart" >
    <mx:series>
        <mx:PieSeries field="Quantidade"
labelPosition="callout" nameField="fonteTipo" >
            <mx:fills>
                <mx:RadialGradient>
                    <mx:entries>
                        <mx:GradientEntry
color="#E9C836" />
                        <mx:GradientEntry
color="#AA9127" />
                    </mx:entries>
                </mx:RadialGradient>
                <mx:RadialGradient>
                    <mx:entries>
                        <mx:GradientEntry
color="#A1AECF" />
                        <mx:GradientEntry
color="#47447A" />
                    </mx:entries>
                </mx:RadialGradient>
                <mx:RadialGradient>
                    <mx:entries>
                        <mx:GradientEntry
color="#339933" />
                        <mx:GradientEntry
color="#339998" />
                    </mx:entries>
                </mx:RadialGradient>
            </mx:fills>
        </mx:PieSeries>
    </mx:series>
</mx:PieChart>
</mx:Box>
<mx:Script source="trafego.as" />
</mx:TitleWindow>

```

7.8.7 Componente Flex DashBoardVisitante_Component.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<mx:TitleWindow title="Visitantes" xmlns:mx="http://www.adobe.com/2006/mxml"
layout="vertical" width="796" height="150"
creationComplete="initApp();" >
    <mx:SeriesInterpolate duration="600" id="showData" />
    <mx:VBox width="750" height="96">
        <mx:LineChart dataProvider="{dadosInputPeriodos}"
showDataTips="true" itemClick="updateDetails(event)" width="750" height="87">
            <mx:horizontalAxis>
                <mx:CategoryAxis categoryField="nome"
dataProvider="{dadosInputPeriodos}" />
            </mx:horizontalAxis>
            <mx:series>
                <mx:LineSeries yField="qtde" displayName="Quantidade"

```

```

showDataEffect="showData" />
    </mx:series>
    </mx:LineChart>
</mx:VBox>
<mx:HTTPService
    id="service"
    url="http://localhost:8080/jsp/requisicao.jsp"
    result="resultHandler(event)"
    fault="faultHandler(event)"
    method="GET" />
    <mx:Script source="visitante.as" />
</mx:TitleWindow>

```

7.8.8 Arquivo ActionScript visitante.as

```

// ActionScript file

import mx.events.CloseEvent;
import mx.managers.PopUpManager;
import mx.charts.events.ChartItemEvent;

import mx.utils.ObjectUtil;
import mx.controls.Alert;
import mx.rpc.events.ResultEvent;
import mx.rpc.events.FaultEvent;
import mx.collections.ArrayCollection;

[Bindable]
var dadosInputVisitantes:ArrayCollection;

[Bindable]
var dadosInputPeriodos:ArrayCollection;

[Bindable]
var flStatus:int;

function initApp():void{
    service.send();
}

function resultHandler(event:ResultEvent):void{
    trace(ObjectUtil.toString(event.result));
    dadosInputVisitantes = event.result.acessos.acao;
    dadosInputPeriodos = dadosInputVisitantes;
    flStatus = 0;
}

function faultHandler(event:FaultEvent):void{
    Alert.show(ObjectUtil.toString(event.fault));
}

function atualizaPeriodoAno():void{
    dadosInputPeriodos = dadosInputVisitantes;
}

```

```

        flStatus = 0;
    }

    function updateDetails(event:ChartItemEvent):void{
        if(flStatus == 0){
            dadosInputPeriodos = event.hitData.item.meses.mes;
            flStatus = 1;
        }else if(flStatus == 1){
            dadosInputPeriodos =
event.hitData.item.semanas.semana;
            flStatus = 2;
        }else if(flStatus == 2){
            dadosInputPeriodos = event.hitData.item.datas.data;
            flStatus = 3;
        }
    }
}

```

7.8.9 Arquivo ActionScript navegador.as

```

// ActionScript file

import mx.collections.ArrayCollection;

[Bindable]
var navegadores:ArrayCollection = new ArrayCollection([{nome:
"Chrome", porcentagem: 20},
{nome: "Firefox", porcentagem: 30}, {nome: "Internet Explorer",
porcentagem: 45},
{nome: "Safar", porcentagem: 5}]);

```

7.8.10 Arquivo ActionScript trafego.xml

```

// ActionScript file

import mx.collections.ArrayCollection;
import mx.charts.events.ChartItemEvent;

[Bindable]
var fonteTrafego:ArrayCollection = new
ArrayCollection([{fonteTipo: "Tráfego Direto", Quantidade: 92},
{fonteTipo: "Mecânismo de Pesquisa", Quantidade: 52}, {fonteTipo:
"Referência", Quantidade: 50}]);

function chartLabel(dataItem:Object, fiel:String, index:int,
dataPercent:Number):String{
    var rounded:Number = Math.round(dataPercent);
}

```

```

        return dataItem.fonteTipo + " representa \n"+ rounded + "%
de fonte de tráfego";
    }

    function chartLabelSemDetalhes(dataItem:Object, fiel:String,
index:int, dataPercent:Number):String{
        var rounded:Number = Math.round(dataPercent);
        return rounded + "% - "+dataItem.fonteTipo;
    }

```

7.9 CÓDIGO FONTE DA PASTA DADOS

7.9.1 Arquivo de dados dadosacessos.xml

```

<acessos>
  <acesso>
    <nome>2007</nome>
    <qtde>300</qtde>
    <meses>
      <mes>
        <nome>Jan</nome>
        <qtde>101</qtde>
        <semanas>
          <semana>
            <nome>Primeira</nome>
            <qtde>15</qtde>
            <datas>
              <data>
                <nome>01/01/2007</nome>
                <qtde>3</qtde>
              </data>
              <data>
                <nome>02/01/2007</nome>
                <qtde>1</qtde>
              </data>
              <data>
                <nome>03/01/2007</nome>
                <qtde>3</qtde>
              </data>
              <data>
                <nome>04/01/2007</nome>
                <qtde>1</qtde>
              </data>
              <data>
                <nome>05/01/2007</nome>
                <qtde>3</qtde>
              </data>
              <data>
                <nome>06/01/2007</nome>
                <qtde>1</qtde>
            </datas>
          </semana>
        </semanas>
      </mes>
    </meses>
  </acesso>
</acessos>

```

```

        </data>
        <data>
        <nome>07/01/2007</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Segunda</nome>
<qtde>20</qtde>
    <datas>
        <data>
        <nome>08/01/2007</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>09/01/2007</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>10/01/2007</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>11/01/2007</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>12/01/2007</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>13/01/2007</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>14/01/2007</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Terceira</nome>
<qtde>25</qtde>
    <datas>
        <data>
        <nome>15/01/2007</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>16/01/2007</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>17/01/2007</nome>
        <qtde>3</qtde>
        </data>
        <data>

```

```

        <nome>18/01/2007</nome>
        <qtde>6</qtde>
      </data>
    <data>
      <nome>19/01/2007</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>20/01/2007</nome>
      <qtde>1</qtde>
    </data>
    <data>
      <nome>21/01/2007</nome>
      <qtde>3</qtde>
    </data>
  </datas>
</semana>
<semana>
  <nome>Quarta</nome>
  <qtde>29</qtde>
  <datas>
    <data>
      <nome>22/01/2007</nome>
      <qtde>8</qtde>
    </data>
    <data>
      <nome>23/01/2007</nome>
      <qtde>1</qtde>
    </data>
    <data>
      <nome>24/01/2007</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>25/01/2007</nome>
      <qtde>6</qtde>
    </data>
    <data>
      <nome>26/01/2007</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>27/01/2007</nome>
      <qtde>5</qtde>
    </data>
    <data>
      <nome>28/01/2007</nome>
      <qtde>3</qtde>
    </data>
  </datas>
</semana>
<semana>
  <nome>Quinta</nome>
  <qtde>12</qtde>
  <datas>
    <data>
      <nome>29/01/2007</nome>
      <qtde>8</qtde>

```

```

</data>
<data>
<nome>30/01/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>31/01/2007</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
</semanas>
</mes>
<mes>
<nome>Fev</nome>
<qtde>121</qtde>
<semanas>
<semana>
<nome>Primeira</nome>
<qtde>15</qtde>
<datas>
<data>
<nome>01/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>02/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>03/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>04/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>05/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>06/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>07/02/2007</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
<semana>
<nome>Segunda</nome>
<qtde>20</qtde>
<datas>
<data>
<nome>08/02/2007</nome>
<qtde>8</qtde>

```



```

</data>
<data>
<nome>09/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>10/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>11/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>12/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>13/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>14/02/2007</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
<semana>
<nome>Terceira</nome>
<qtde>25</qtde>
<datas>
<data>
<nome>15/02/2007</nome>
<qtde>8</qtde>
</data>
<data>
<nome>16/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>17/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>18/02/2007</nome>
<qtde>6</qtde>
</data>
<data>
<nome>19/02/2007</nome>
<qtde>3</qtde>
</data>
<data>
<nome>20/02/2007</nome>
<qtde>1</qtde>
</data>
<data>
<nome>21/02/2007</nome>
<qtde>3</qtde>

```

```

        </data>
      </datas>
    </semana>
    <semana>
    <nome>Quarta</nome>
    <qtde>61</qtde>
    <datas>
      <data>
        <nome>22/02/2007</nome>
        <qtde>8</qtde>
      </data>
      <data>
        <nome>23/02/2007</nome>
        <qtde>33</qtde>
      </data>
      <data>
        <nome>24/02/2007</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>25/02/2007</nome>
        <qtde>6</qtde>
      </data>
      <data>
        <nome>26/02/2007</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>27/02/2007</nome>
        <qtde>5</qtde>
      </data>
      <data>
        <nome>28/02/2007</nome>
        <qtde>3</qtde>
      </data>
    </datas>
  </semana>
</semanas>
</mes>
</meses>
</acesso>
<acesso>
<nome>2008</nome>
<qtde>312</qtde>
<meses>
  <mes>
    <nome>Jan</nome>
    <qtde>101</qtde>
    <semanas>
      <semana>
        <nome>Primeira</nome>
        <qtde>15</qtde>
        <datas>
          <data>
            <nome>01/01/2008</nome>
            <qtde>3</qtde>
          </data>
          <data>

```

```

<nome>02/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>03/01/2008</nome>
<qtde>3</qtde>
</data>
<data>
<nome>04/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>05/01/2008</nome>
<qtde>3</qtde>
</data>
<data>
<nome>06/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>07/01/2008</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
<semana>
<nome>Segunda</nome>
<qtde>20</qtde>
<datas>
<data>
<nome>08/01/2008</nome>
<qtde>8</qtde>
</data>
<data>
<nome>09/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>10/01/2008</nome>
<qtde>3</qtde>
</data>
<data>
<nome>11/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>12/01/2008</nome>
<qtde>3</qtde>
</data>
<data>
<nome>13/01/2008</nome>
<qtde>1</qtde>
</data>
<data>
<nome>14/01/2008</nome>
<qtde>3</qtde>
</data>
</datas>

```

```
</semana>
<semana>
<nome>Terceira</nome>
<qtde>25</qtde>
  <datas>
    <data>
      <nome>15/01/2008</nome>
      <qtde>8</qtde>
    </data>
    <data>
      <nome>16/01/2008</nome>
      <qtde>1</qtde>
    </data>
    <data>
      <nome>17/01/2008</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>18/01/2008</nome>
      <qtde>6</qtde>
    </data>
    <data>
      <nome>19/01/2008</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>20/01/2008</nome>
      <qtde>1</qtde>
    </data>
    <data>
      <nome>21/01/2008</nome>
      <qtde>3</qtde>
    </data>
  </datas>
</semana>
<semana>
<nome>Quarta</nome>
<qtde>29</qtde>
  <datas>
    <data>
      <nome>22/01/2008</nome>
      <qtde>8</qtde>
    </data>
    <data>
      <nome>23/01/2008</nome>
      <qtde>1</qtde>
    </data>
    <data>
      <nome>24/01/2008</nome>
      <qtde>3</qtde>
    </data>
    <data>
      <nome>25/01/2008</nome>
      <qtde>6</qtde>
    </data>
    <data>
      <nome>26/01/2008</nome>
      <qtde>3</qtde>
    </data>
  </datas>
</semana>
```

```

        </data>
        <data>
        <nome>27/01/2008</nome>
        <qtde>5</qtde>
        </data>
        <data>
        <nome>28/01/2008</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Quinta</nome>
<qtde>12</qtde>
    <datas>
        <data>
        <nome>29/01/2008</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>30/01/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>31/01/2008</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
</semanas>
</mes>
<mes>
<nome>Fev</nome>
<qtde>121</qtde>
    <semanas>
        <semana>
        <nome>Primeira</nome>
        <qtde>15</qtde>
        <datas>
            <data>
            <nome>01/02/2008</nome>
            <qtde>3</qtde>
            </data>
            <data>
            <nome>02/02/2008</nome>
            <qtde>1</qtde>
            </data>
            <data>
            <nome>03/02/2008</nome>
            <qtde>3</qtde>
            </data>
            <data>
            <nome>04/02/2008</nome>
            <qtde>1</qtde>
            </data>
            <data>
            <nome>05/02/2008</nome>
            <qtde>3</qtde>

```

```

        </data>
        <data>
        <nome>06/02/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>07/02/2008</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Segunda</nome>
<qtde>20</qtde>
    <datas>
        <data>
        <nome>08/02/2008</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>09/02/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>10/02/2008</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>11/02/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>12/02/2008</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>13/02/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>14/02/2008</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Terceira</nome>
<qtde>25</qtde>
    <datas>
        <data>
        <nome>15/02/2008</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>16/02/2008</nome>
        <qtde>1</qtde>
        </data>
        <data>

```

```

        <nome>17/02/2008</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>18/02/2008</nome>
        <qtde>6</qtde>
      </data>
      <data>
        <nome>19/02/2008</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>20/02/2008</nome>
        <qtde>1</qtde>
      </data>
      <data>
        <nome>21/02/2008</nome>
        <qtde>3</qtde>
      </data>
    </datas>
  </semana>
  <semana>
    <nome>Quarta</nome>
    <qtde>61</qtde>
    <datas>
      <data>
        <nome>22/02/2008</nome>
        <qtde>8</qtde>
      </data>
      <data>
        <nome>23/02/2008</nome>
        <qtde>33</qtde>
      </data>
      <data>
        <nome>24/02/2008</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>25/02/2008</nome>
        <qtde>6</qtde>
      </data>
      <data>
        <nome>26/02/2008</nome>
        <qtde>3</qtde>
      </data>
      <data>
        <nome>27/02/2008</nome>
        <qtde>5</qtde>
      </data>
      <data>
        <nome>28/02/2008</nome>
        <qtde>3</qtde>
      </data>
    </datas>
  </semana>
</semanas>
</mes>
</meses>

```

```

</acesso>
<acesso>
<nome>2009</nome>
<qtde>343</qtde>
  <meses>
    <mes>
      <nome>Jan</nome>
      <qtde>101</qtde>
      <semanas>
        <semana>
          <nome>Primeira</nome>
          <qtde>15</qtde>
          <datas>
            <data>
              <nome>01/01/2009</nome>
              <qtde>3</qtde>
            </data>
            <data>
              <nome>02/01/2009</nome>
              <qtde>1</qtde>
            </data>
            <data>
              <nome>03/01/2009</nome>
              <qtde>3</qtde>
            </data>
            <data>
              <nome>04/01/2009</nome>
              <qtde>1</qtde>
            </data>
            <data>
              <nome>05/01/2009</nome>
              <qtde>3</qtde>
            </data>
            <data>
              <nome>06/01/2009</nome>
              <qtde>1</qtde>
            </data>
            <data>
              <nome>07/01/2009</nome>
              <qtde>3</qtde>
            </data>
          </datas>
        </semana>
        <semana>
          <nome>Segunda</nome>
          <qtde>20</qtde>
          <datas>
            <data>
              <nome>08/01/2009</nome>
              <qtde>8</qtde>
            </data>
            <data>
              <nome>09/01/2009</nome>
              <qtde>1</qtde>
            </data>
            <data>
              <nome>10/01/2009</nome>
              <qtde>3</qtde>
            </data>
          </datas>
        </semana>
      </semanas>
    </mes>
  </meses>
</acesso>

```



```

        </data>
        <data>
        <nome>11/01/2009</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>12/01/2009</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>13/01/2009</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>14/01/2009</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Terceira</nome>
<qtde>25</qtde>
    <datas>
        <data>
        <nome>15/01/2009</nome>
        <qtde>8</qtde>
        </data>
        <data>
        <nome>16/01/2009</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>17/01/2009</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>18/01/2009</nome>
        <qtde>6</qtde>
        </data>
        <data>
        <nome>19/01/2009</nome>
        <qtde>3</qtde>
        </data>
        <data>
        <nome>20/01/2009</nome>
        <qtde>1</qtde>
        </data>
        <data>
        <nome>21/01/2009</nome>
        <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Quarta</nome>
<qtde>29</qtde>
    <datas>
        <data>

```

```

                <nome>22/01/2009</nome>
                <qtde>8</qtde>
            </data>
            <data>
                <nome>23/01/2009</nome>
                <qtde>1</qtde>
            </data>
            <data>
                <nome>24/01/2009</nome>
                <qtde>3</qtde>
            </data>
            <data>
                <nome>25/01/2009</nome>
                <qtde>6</qtde>
            </data>
            <data>
                <nome>26/01/2009</nome>
                <qtde>3</qtde>
            </data>
            <data>
                <nome>27/01/2009</nome>
                <qtde>5</qtde>
            </data>
            <data>
                <nome>28/01/2009</nome>
                <qtde>3</qtde>
            </data>
        </datas>
    </semana>
    <semana>
        <nome>Quinta</nome>
        <qtde>12</qtde>
        <datas>
            <data>
                <nome>29/01/2009</nome>
                <qtde>8</qtde>
            </data>
            <data>
                <nome>30/01/2009</nome>
                <qtde>1</qtde>
            </data>
            <data>
                <nome>31/01/2009</nome>
                <qtde>3</qtde>
            </data>
        </datas>
    </semana>
</semanas>
</mes>
<mes>
    <nome>Fev</nome>
    <qtde>121</qtde>
    <semanas>
        <semana>
            <nome>Primeira</nome>
            <qtde>15</qtde>
            <datas>
                <data>

```

```

<nome>01/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>02/02/2009</nome>
<qtde>1</qtde>
</data>
<data>
<nome>03/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>04/02/2009</nome>
<qtde>1</qtde>
</data>
<data>
<nome>05/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>06/02/2009</nome>
<qtde>1</qtde>
</data>
<data>
<nome>07/02/2009</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
<semana>
<nome>Segunda</nome>
<qtde>20</qtde>
<datas>
<data>
<nome>08/02/2009</nome>
<qtde>8</qtde>
</data>
<data>
<nome>09/02/2009</nome>
<qtde>1</qtde>
</data>
<data>
<nome>10/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>11/02/2009</nome>
<qtde>1</qtde>
</data>
<data>
<nome>12/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>13/02/2009</nome>
<qtde>1</qtde>
</data>
<data>

```

```

                <nome>14/02/2009</nome>
                <qtde>3</qtde>
            </data>
        </datas>
    </semana>
    <semana>
    <nome>Terceira</nome>
    <qtde>25</qtde>
    <datas>
        <data>
            <nome>15/02/2009</nome>
            <qtde>8</qtde>
        </data>
        <data>
            <nome>16/02/2009</nome>
            <qtde>1</qtde>
        </data>
        <data>
            <nome>17/02/2009</nome>
            <qtde>3</qtde>
        </data>
        <data>
            <nome>18/02/2009</nome>
            <qtde>6</qtde>
        </data>
        <data>
            <nome>19/02/2009</nome>
            <qtde>3</qtde>
        </data>
        <data>
            <nome>20/02/2009</nome>
            <qtde>1</qtde>
        </data>
        <data>
            <nome>21/02/2009</nome>
            <qtde>3</qtde>
        </data>
    </datas>
</semana>
<semana>
<nome>Quarta</nome>
<qtde>61</qtde>
    <datas>
        <data>
            <nome>22/02/2009</nome>
            <qtde>8</qtde>
        </data>
        <data>
            <nome>23/02/2009</nome>
            <qtde>33</qtde>
        </data>
        <data>
            <nome>24/02/2009</nome>
            <qtde>3</qtde>
        </data>
        <data>
            <nome>25/02/2009</nome>
            <qtde>6</qtde>
    </datas>

```

```

</data>
<data>
<nome>26/02/2009</nome>
<qtde>3</qtde>
</data>
<data>
<nome>27/02/2009</nome>
<qtde>5</qtde>
</data>
<data>
<nome>28/02/2009</nome>
<qtde>3</qtde>
</data>
</datas>
</semana>
</semanas>
</mes>
</meses>
</acesso>
</acessos>

```

7.9.2 Arquivo de dados dadosnavegadoresefontestrafego.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<estatisticas>
  <dia>
    <data>2009-09-10</data>
    <acessos>20</acessos>
    <fontestrafego>
      <fontetrafeço>
        <tipo>direto</tipo>
        <acessos>
          9
        </acessos>
      </fontetrafeço>
      <fontetrafeço>
        <tipo>
          pesquisa
        </tipo>
        <acessos>
          6
        </acessos>
      </fontetrafeço>
      <fontetrafeço>
        <tipo>
          referencia
        </tipo>
        <acessos>
          5
        </acessos>
      </fontetrafeço>
    </fontestrafego>
  </dia>

```

```

<navegadores>
  <navegador>
    <nome>internet explorer</nome>
    <acessos>15</acessos>
  </navegador>
  <navegador>
    <nome>firefox</nome>
    <acessos>4</acessos>
  </navegador>
  <navegador>
    <nome>chrome</nome>
    <acessos>1</acessos>
  </navegador>
  <navegador>
    <nome>outros</nome>
    <acessos>0</acessos>
  </navegador>
</navegadores>
</dia>
<dia>
  <data>2009-09-11</data>
  <acessos>30</acessos>
  <fontestrafego>
    <fontetrafeço>
      <tipo>
        direto
      </tipo>
      <acessos>
        14
      </acessos>
    </fontetrafeço>
    <fontetrafeço>
      <tipo>
        pesquisa
      </tipo>
      <acessos>
        9
      </acessos>
    </fontetrafeço>
    <fontetrafeço>
      <tipo>
        referencia
      </tipo>
      <acessos>
        8
      </acessos>
    </fontetrafeço>
  </fontestrafego>
</navegadores>
  <navegador>
    <nome>internet explorer</nome>
    <acessos>25</acessos>
  </navegador>
  <navegador>
    <nome>firefox</nome>
    <acessos>4</acessos>
  </navegador>
</navegador>

```

```

        <nome>chrome</nome>
        <acessos>1</acessos>
    </navegador>
    <navegador>
        <nome>outros</nome>
        <acessos>0</acessos>
    </navegador>
</navegadores>
</dia>
<dia>
    <data>200-09-12</data>
    <acessos>40</acessos>
    <fontestrafego>
        <fontetrafeço>
            <tipo>
                direto
            </tipo>
            <acessos>
                38
            </acessos>
        </fontetrafeço>
        <fontetrafeço>
            <tipo>
                pesquisa
            </tipo>
            <acessos>
                12
            </acessos>
        </fontetrafeço>
        <fontetrafeço>
            <tipo>
                referencia
            </tipo>
            <acessos>
                10
            </acessos>
        </fontetrafeço>
    </fontestrafego>
    <navegadores>
        <navegador>
            <nome>internet explorer</nome>
            <acessos>30</acessos>
        </navegador>
        <navegador>
            <nome>firefox</nome>
            <acessos>6</acessos>
        </navegador>
        <navegador>
            <nome>chrome</nome>
            <acessos>3</acessos>
        </navegador>
        <navegador>
            <nome>outros</nome>
            <acessos>1</acessos>
        </navegador>
    </navegadores>
</dia>
</estatisticas>

```

7.10 CÓDIGO FONTE DA PASTA DE CONFIGURAÇÃO META-INF

7.10.1 Arquivo de configuração application.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE application PUBLIC
    "-//Sun Microsystems, Inc.//DTD J2EE Application 1.3//EN"
    "http://java.sun.com/dtd/application_1_3.dtd">
<application>
  <display-name>TccApp</display-name>
  <description>Aplicação do TCC</description>
  <module>
    <web>
      <web-uri>war</web-uri>
      <context-root></context-root>
    </web>
  </module>
  <module>
    <java>TccApp.jar</java>
  </module>
  <module>
    <java>TccAppFlex.jar</java>
  </module>
  <module>
    <ejb>TccAppEjb.jar</ejb>
  </module>
</application>
```

7.10.2 Arquivo de configuração jboss-app.xml

```
<jboss-app>
  <loader-repository>
    TccApp:archive=tccapp.ear
  </loader-repository>
</jboss-app>
```

7.11 CÓDIGO FONTE DA PASTA DO CONTEXTO WEB WAR

7.11.1 Arquivo de requisição Flex requisicao.jsp

```

<%@ page contentType="text/xml"
import="br.ufsc.inf.tccapp.flex"
%>

<%
String method = request.getParameter("method");

TCC_APPSeervlet controller = new TCC_APPSeervlet();

if (method.equals("getPaises")){
%>
<%= controller.getPaises() %>
<%
}
%>

```

7.11.2 Arquivo de configuração appengine-web.xml

```

<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application>skeletomagrelotccapp</application>
  <version>1</version>

  <!-- Configure java.util.logging -->
  <system-properties>
    <property name="java.util.logging.config.file" value="WEB-
INF/logging.properties"/>
  </system-properties>
</appengine-web-app>

```

7.11.3 Arquivo de configuração logging.properties

```

# A default java.util.logging configuration.
# (All App Engine logging is through java.util.logging by default).
#
# To use this configuration, copy it into your application's WEB-INF
# folder and add the following to your appengine-web.xml:
#
# <system-properties>
#   <property name="java.util.logging.config.file" value="WEB-
INF/logging.properties"/>
# </system-properties>

```

```
#
# Set the default logging level for all loggers to WARNING
.level = WARNING

# Set the default logging level for ORM, specifically, to WARNING
DataNucleus.JDO.level=WARNING
DataNucleus.Persistence.level=WARNING
DataNucleus.Cache.level=WARNING
DataNucleus.MetaData.level=WARNING
DataNucleus.General.level=WARNING
DataNucleus.Utility.level=WARNING
DataNucleus.Transaction.level=WARNING
DataNucleus.Datastore.level=WARNING
DataNucleus.ClassLoading.level=WARNING
DataNucleus.Plugin.level=WARNING
DataNucleus.ValueGeneration.level=WARNING
DataNucleus.Enhancer.level=WARNING
DataNucleus.SchemaTool.level=WARNING
```

7.11.4 Arquivo de configuração web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

  <!-- Servlets -->
  <!-- GWT Services -->
  <servlet>
    <servlet-name>visitorsChartDataServiceServlet</servlet-name>
    <servlet-
class>br.ufsc.inf.tccapp.gwt.server.services.VisitorsChartDataServiceImpl</serv
vlet-class>
    </servlet>
    <servlet-mapping>
      <servlet-name>visitorsChartDataServiceServlet</servlet-name>
      <url-pattern>/tccapp/visitorschartdataservice</url-pattern>
    </servlet-mapping>
    <servlet>
      <servlet-name>trafficChartDataServiceServlet</servlet-name>
      <servlet-
class>br.ufsc.inf.tccapp.gwt.server.services.TrafficChartDataServiceImpl</serv
let-class>
      </servlet>
      <servlet-mapping>
        <servlet-name>trafficChartDataServiceServlet</servlet-name>
        <url-pattern>/tccapp/trafficchartdataservice</url-pattern>
      </servlet-mapping>
    <servlet>
      <servlet-name>browsersChartDataServiceServlet</servlet-name>
```

```

    <servlet-
class>br.ufsc.inf.tccapp.gwt.server.services.BrowsersChartDataServiceImpl</ser
vlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>browsersChartDataServiceServlet</servlet-name>
        <url-pattern>/tccapp/browserschartdataservice</url-pattern>
    </servlet-mapping>
    <!-- Flex Servlets -->
    <servlet>
        <servlet-name>TCC_APP</servlet-name>
        <servlet-class>br.ufsc.inf.tccapp.flex.TCC_APPServlet</servlet-
class>
    </servlet>
    <servlet-mapping>
        <servlet-name>TCC_APP</servlet-name>
        <url-pattern>/jsp/requisicao.jsp</url-pattern>
    </servlet-mapping>

    <!-- Default page to serve -->
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>TCC_Application.swf</welcome-file>
        <welcome-file>TccApp.html</welcome-file>
    </welcome-file-list>

</web-app>

```

7.11.5 Arquivo de interface inicial index.html

```

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>RIApplication</title>
<style>
    body {
        font-family: tahoma, arial;
        font-weight: bold;
        margin: 0 auto;
        text-align: center;
        background-color: #CFEEFF;
    }

    a {
        text-decoration: none;
        color: inherit;
    }

    #logo{
        margin-top: 50px;

```

```

    }

    #frase {
        margin: 50px 0 25px 0;
    }

    #links {
        font-size: 3em;
    }

    #links a{
        margin: 50px;
        color:black;
    }
</style>
</head>
<body>
    
    <div id="frase">Escolha a Tecnologia:</div>
    <div id="links">
        <a href="TCC_Application.swf">Adobe Flex</a>
        <a href="TccApp.html">GWT-EXT</a>
    </div>
</body>
</html>

```

7.11.6 Arquivo de estilo TccApp.css

```

/** Add css rules here for your application. */

/** Most GWT widgets already have a style name defined */
.gwt-DialogBox {
    width: 400px;
}

.dialogVPanel {
    margin: 5px;
}

.serverResponseLabelError {
    color: red;
}

/** Set ids using widget.getElement().setId("idOfElement") */
#closeButton {
    margin: 15px 6px 6px;
}

```

7.11.7 Arquivo de interface GWT-EXT TccApp.html

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- The HTML 4.01 Transitional DOCTYPE declaration-->
<!-- above set at the top of the file will set -->
<!-- the browser's rendering engine into -->
<!-- "Quirks Mode". Replacing this declaration -->
<!-- with a "Standards Mode" doctype is supported, -->
<!-- but may lead to some differences in layout. -->

<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">

    <!-- -->
    <!-- Consider inlining CSS to reduce the number of requested files -->
    <!-- -->
    <link type="text/css" rel="stylesheet" href="TccApp.css">

    <!-- -->
    <!-- Any title is fine -->
    <!-- -->
    <title>RIApplication</title>

    <!-- -->
    <!-- This script loads your compiled module. -->
    <!-- If you add any GWT meta tags, they must -->
    <!-- be added before this line. -->
    <!-- -->
    <script type="text/javascript" language="javascript"
src="tccapp/tccapp.nocache.js"></script>
  </head>

  <!-- -->
  <!-- The body can have arbitrary html, or -->
  <!-- you can leave the body empty if you want -->
  <!-- to create a completely dynamic UI. -->
  <!-- -->
  <body>

    <!-- OPTIONAL: include this if you want history support -->
    <iframe src="javascript:''" id="__gwt_historyFrame" tabIndex='-1'
style="position:absolute;width:0;height:0;border:0"></iframe>

  </body>
</html>

```