CENTRO UNIVERSITÁRIO FEEVALE

DIOGO RAFAEL JACOBS

UM GERENCIADOR DE RELATÓRIOS UTILIZANDO AJAX

Novo Hamburgo, novembro de 2006.

DIOGO RAFAEL JACOBS

UM GERENCIADOR DE RELATÓRIOS UTILIZANDO AJAX

Centro Universitário Feevale
Instituto de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação
Trabalho de Conclusão de Curso

Professor orientador: Juliano Varella de Carvalho

Novo Hamburgo, novembro de 2006.

AGRADECIMENTOS

Gostaria de agradecer a todas as pessoas que de alguma forma contribuíram para o desenvolvimento desse trabalho:

Primeiramente a Deus, por tornar possível mais essa conquista.

A minha família que sempre me apoiou, mesmo nas horas mais difíceis, durante essa longa caminhada.

Ao meu orientador, Juliano Varella de Carvalho, pelo companheirismo, conhecimento e dedicação que foram fundamentais durante todo o trabalho.

Aos colegas da Feevale, especialmente a Silvia Berghan, pela ajuda dispensada.

“Um dia você aprende que...não importa onde já chegou, mas onde está indo..”(Willian Shakespeare)

RESUMO

Atualmente, o mundo corporativo caracterizado pelo alto grau de competitividade, exige das empresas agilidade na obtenção de informações acerca de um assunto ou área de interesse. Ter a informação é importante, porém não é mais um fator isolado, que leva ao sucesso e ao destaque em relação aos concorrentes. Uma vez que o uso de sistemas de banco de dados para o armazenamento e organização dessas informações já é um consenso entre grande parte das empresas e vêm sendo utilizado em larga escala, o diferencial competitivo pode estar na agilidade e precisão na obtenção das informações. O uso de gerenciadores de relatórios para auxiliar na extração de informações vem crescendo, porém são ferramentas que normalmente exigem um certo conhecimento técnico por parte dos usuários. A popularidade do uso da Internet vem ganhando força no mundo moderno. O surgimento de tecnologias inovadoras que viabilizam o desenvolvimento de aplicações e ferramentas complexas sob ambiente Web, está trazendo uma nova realidade, onde uma completa gama de utilitários podem ser disponibilizados em forma de aplicações interpretadas e executadas em um navegador Web, substituindo as ferramentas *desktop*. A cada dia surge uma nova ferramenta trabalhando sob a plataforma Web, como por exemplo: processadores de texto, planilhas eletrônicas, agenda pessoal, cliente de email, entre outras. Sendo assim, neste trabalho, pretende-se apresentar as inovações que surgiram no que tange ao desenvolvimento Web, as dificuldades encontradas nos sistemas de informação atuais, bem como propor o desenvolvimento de um gerenciador de relatórios sob plataforma Web, a fim de tornar a tarefa de criação de relatórios mais dinâmica, flexível e amigável ao usuário.

Palavras-chave: Extração de Informações, Relatórios, Internet, RIA, AJAX.

ABSTRACT

Today's highly competitive corporate world demands from enterprises a lot of agility in obtaining information about subjects or areas in their interest. Merely having the information is important, although it's no longer an isolated factor in the path to success and differentiation from competitors. Given that the use of database systems to store and organize such heaps of information is already a consensus among most corporations and has been put to use in large scale, competitive differentiation can be found in the agility and precision with which that information is retrieved from the databases. The use of report generators as an aid to information retrieval has been a growing trend recently, yet those tools demand a certain level of technical knowledge from their users. The popularity of the Internet has been getting stronger every day in the modern world. The unveiling of innovative technologies that foster the development of complex web applications and tools is bringing about a new reality, where a complete range of utilities can be provided in the form of applications that run on a web browser, easily replacing desktop tools. Every day a new tool running on a web environment is made available, prividing all kinds of functionality, such as: word processing, electronic spreadsheets, personal scheduling, email clients and so on. Given this context, this work intends to present the innovations that showed up in the Web development world, the difficulties found in current information systems, as well as propose the development of a report generator over the Web platform, in order to turn report creation into a more dynamic, flexible and user-friendly task.

Palavras-chave: Information Retrieval, Reports, Internet, RIA, AJAX.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AJAX	*Asynchronous Javascript And XML*
API	*Application Programming Interface*
BI	*Business Intelligence*
CSS	*Cascade Style Sheets*
CSV	*Comma Separated Value*
DWR	*Direct Web Remoting*
ERP	*Enterprise Resource Planning*
FTP	*File Transfer Protocol*
GUI	*Graphical User Interface*
IDE	*Integrated Development Environment*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
MIT	*Massachussets Institute of Technology*
NCSA	*Nacional Center for Supercomputing Applications*
NLS	*OnLine System*
PDA	*Personal Digital Assistence*
PHP	*Hypertext Preprocessor*
RIA	*Rich Internet Application*
ROI	*Return On Investment*
SGBD	Sistema de Gerenciamento de Banco de Dados
SLAC	*Stanford Linear Accelerator Center*
SMTP	*Simple Mail Transfer Protocol*
SOA	*Service Oriented Architecture*
SQL	*Structured Query Language*

SSH	*Secure SHell*
SWF	*ShockWave Flash*
TXT	*TeXT*
URL	*Universal Resource Locator*
XHTML	*eXtensible Hypertext Markup Language*
XML	*eXtensible Markup Language*
W3C	*World Wide Web Consortium*
WYSIWYG	*What You See Is What You Get*

SUMÁRIO

INTRODUÇÃO

O mundo corporativo atual, caracterizado principalmente pelo alto grau de competitividade entre as organizações que o forma, busca-se cada vez mais diferenciais competitivos que podem fortalecer a empresa, fazendo com que ela se destaque em relação as demais. Além disso, a extensão dessa competitividade em âmbito mundial, faz com que a tomada de decisão no momento certo leve ao sucesso ou fracasso de uma organização. Sendo assim, ter acesso à informação não pode mais ser encarado como exclusivo diferencial competitivo, uma vez que, informações na Internet estão acessíveis para qualquer um e em abundância. Esse novo contexto exige das empresas novas estratégias e meios para conseguir essa informação de forma precisa e em tempo hábil.

A constante evolução da tecnologia tem gerado situações onde o acesso rápido, preciso e fácil a informação é de fundamental importância para o sucesso das tarefas cotidianas. Soluções de armazenamento, classificação e organização de dados vêm sendo utilizadas para auxiliar as empresas no desafio diário de absorver a imensa quantidade de informação. Soluções de sistemas de bancos de dados já são consagradas e utilizadas em grande parte dessas organizações. Dado esse fato, o meio de armazenar tais informações já é conhecido, e utilizado em larga escala por uma grande maioria.

Silberschartz, Korth e Sudarshan (1999, p.1) descreve:

> A importância da informação na maioria das organizações – que estabelece o valor do banco de dados – tem determinado o desenvolvimento de um grande conjunto de conceitos e técnicas para a administração eficaz desses valores [...]

Nesse contexto, onde a informação tem um valor importante para todo e qualquer serviço prestado, a organização se torna fundamental para alcançar o alto nível de qualidade e agilidade que o mercado exige. O uso de tecnologias de armazenamento, catalogação e organização dessas informações vêm crescendo, tornando-se um dos pontos chaves na estratégia tecnológica adotada por pequenas e grandes corporações. Uma das opções que permite essa organização é a utilização de sistemas de banco de dados.

Segundo Elmasri e Navathe (2005, p.3), os sistemas de banco de dados tornaram-se essenciais nas aplicações da sociedade moderna. Isto comprova cada vez mais que é difícil imaginar como seria o mundo se não fossem os sistemas de banco de dados. Essa popularização dá-se principalmente devido a forte utilização de sistemas e técnicas de BI *(Business Intelligence)*, ao forte crescimento de aplicações de comércio eletrônico, que gera necessidade de integração entre sistemas, e a todo o conjunto de sistemas legados já existentes.

Sendo assim, um diferencial competitivo pode ser a maneira com que esses dados são extraídos e disponibilizados em um formato que seres humanos podem avaliar e compreender. Um exemplo disso são os relatórios, que a partir de uma consulta e/ou cruzamento de informações extraídas de um ou mais sistemas de banco de dados, formatam, tratam e exibem de uma maneira mais formal essas informações de modo a agregar valor, permitindo que o dado que estava armazenado se torne algo importante que pode auxiliar na tomada de decisões.

Essas necessidades diversas, anteriormente citadas, acabam formando um ambiente misto no que tange aos SGBDs utilizados. Tem-se um novo problema, lidar com esse ambiente, tornando possível a aquisição de informações especificas de cada solução, para viabilizar a extração de informações de diferentes fontes de dados. Por exemplo: o funcionário da empresa X precisa da relação de todos os clientes que compraram mais de mil reais no ano passado, mas que este ano não efetuou movimentações, sendo que a empresa X tem um sistema de banco de dados para as vendas na loja e outro sistema para as vendas no site *eCommerce*[1].

[1] As diferentes bases de dados, são sincronizadas, com as informações de produtos, com o objetivo de controlar o estoque, mas cada uma delas possui dados específicos em relação ao ambiente.

O funcionário tem dois caminhos a seguir: analisar individual e manualmente os extratos de cada cliente, nos dois ambientes; ou, solicitar a criação de uma nova funcionalidade que deverá ser agregada ao software para o fornecedor do mesmo, porém nenhuma das duas situações é favorável. Na primeira, o funcionário levaria muito tempo para conseguir a informação desejada. Na segunda, a demanda gerada para o fornecedor do sistema geraria custos não previstos para a empresa.

Nesse cenário, torna-se comum o uso de sistemas gerenciadores de relatórios, uma opção interessante para quem busca uma maior facilidade e flexibilidade na extração e exibição de informações de um banco de dados. Existem conceituadas ferramentas desse tipo, onde se destacam: Microsoft SQLServer™ SQL Reporting Services, Crystal Report, Active Report, Quick Report. Tais ferramentas criam um relatório que a partir de um *result set*[2]. Porém, a maioria delas são concebidas com um foco diferente, servindo de extensão e auxiliando o desenvolvimento dos sistemas, impossibilitando que usuários menos experientes e não técnicos consigam fazer uso dessa ferramenta. Outro ponto que pode ser considerado negativo é o fato de serem ferramentas que trabalham em ambiente *desktop*, não permitindo a acessibilidade de maneira simples e clara, como uma aplicação Web.

Considerando o cenário apresentado, este trabalho apresenta o estudo e o desenvolvimento de uma ferramenta capaz de criar e visualizar relatórios. A ferramenta proposta ficará disponível em forma de aplicação Web, ou seja, acessível pelo navegador, oferecendo compatibilidade entre os principais navegadores utilizados atualmente, através do uso de padrões Web (*Web Standards*).

Esta ferramenta será concebida utilizando as técnicas oferecidas pela API AJAX, viabilizando uma interação mais rica com o usuário. Os objetivos principais da ferramenta são:

- Conectividade com diferentes bancos de dados, viabilizando a criação de relatórios em ambientes mistos;

- Criação e visualização de relatórios em um ambiente Web;

[2] Refere-se a estrutura retornada por uma consulta de um banco de dados, as informações são retornadas em conjuntos de dados, denominados *result sets*.

- Interface intuitiva de modo, a oferecer recursos para elaboração de relatórios por usuários finais sem necessidade de conhecimento específico.

O presente trabalho é composto de quatro capítulos. No primeiro e segundo capítulos, serão mostradas as evoluções do desenvolvimento Web desde seu surgimento e como o uso do AJAX vem revolucionando as aplicações. No terceiro capitulo será apresentada a importância dos relatórios para as empresas e a carência de uma maior flexibilidade para a concepção de relatórios. No quarto capitulo será apresentado o estudo realizado para a elaboração da ferramenta, apresentando como serão solucionados os problemas levantados na sessão anterior.

1 A EVOLUÇÃO DAS APLICAÇÕES WEB

O objetivo deste capítulo é apresentar o histórico da Web, mostrando a situação de cada momento, como tudo começou e também quais foram as principais inovações até se chegar à Web atual, caracterizada por um cenário, composto de aplicações que até pouco tempo eram possíveis apenas em ambiente *desktop*.

1.1 Conceitos

É comum confundir os termos Web, World Wide Web, WWW e W3 com "Internet", enquanto estes correspondem a um tipo de serviço de troca de informações por meio de hipertexto[3], a Internet é uma rede constituída por um conjunto de outras redes, que se comunicam e colaboram entre si.

> Ao contrário do que muitos pensam, a Internet e a *Word Wide Web* não são sinônimos: a Internet é um conjunto de redes de computadores interconectados, ligados através de fios de cobre, cabos de fibra ótica, conexões *wireless* etc; a *Web* é um conjunto de documentos interligados, ligados através de *hyperlinks* e URLs, e é acessível usando a Internet. (WIKIPEDIA,2006a,p.1)

[3] Sistema que permite a visualização de informação, onde os documentos podem referenciar outros documentos, permitindo criar associações entre informações de um mesmo assunto. Maiores informações em: http://pt.wikipedia.org/wiki/Hipertexto

1.2 Histórico

A criação da World Wide Web tem suas raízes em vários fatos do passado, onde se destacam: a criação do telégrafo, os estudos que originaram o que hoje é conhecido como hipertexto, o lançamento do primeiro satélite artificial pela União Soviética, entre outros.

1.2.1 O hipertexto

O hipertexto é a capacidade de criar associações entre informações e documentos distintos através da criação de ligações, também chamadas *hyperlinks*. Tem suas raízes nos estudos do professor e cientista Vannevar Bush, criador do Memex (*Memory Extension*), criado na década de 30. Tratava-se de um dispositivo mecânico capaz de criar e seguir associações, viabilizando assim o armazenamento de uma quantidade grande de informações e principalmente a localização rápida delas. Maiores informações sobre a invenção podem ser encontradas no artigo "*As We May Think*"[4], publicado em 1945.

> Vannevar Bush nunca foi diretamente envolvido com a criação ou desenvolvimento da Internet. Ele morreu antes da criação da *Word Wide Web*. Mesmo assim muitos consideram Bush, o avô da nossa WIRED AGE freqüentemente fazendo referência aos estudos de 1945, "*As We May Think"*. Nesse artigo, Bush descreveu uma máquina teórica que ele chamou "Memex" com o objetivo de melhorar a memória humana permitindo que o usuário armazenasse e restaurasse documentos ligados por associações. INTERNET(2006,p.1)

Por definição, *World Wide Web* significa teia do tamanho do mundo, seria a tradução literal para o português, conforme é relatado em WIKIPEDIA (2006c,p.1).

A Figura 1-1 representa graficamente a capacidade de referenciar outros documentos através do uso do hipertexto.

[4] O artigo está disponível na Internet no endereço: http://www.theatlantic.com/doc/194507/bush

Figura 1-1 Interligação de documentos

Fonte: http://www.estudar.org/pessoa/internet/02www/people-tim_berners_lee.html

1.2.2 A criação da World Wide Web

Timothy J. Berners-Lee, também conhecido como Tim Berners-Lee, hoje diretor do W3C, é conhecido como o criador da World Wide Web. Auxiliado por Robert Cailliau, criou um protótipo que foi chamado de *ENQUIRE*[5]. O ENQUIRE era um protótipo com algumas das principais funcionalidades da Web e principalmente dos WIKIS[6], muito utilizados nos dias de hoje, e destacava-se pelo uso de banco de dados, criação de relacionamentos através de hyperlinks bidirecionais e a possibilidade de edição direta no servidor

No centro de pesquisa do CERN um dos objetivos era o compartilhamento de informações entre os cientistas que estavam fora do instituto e até mesmo fora do país, os quais enfrentavam sérios problemas para viabilizar a troca de conhecimento, dificultando o trabalho.

[5] Um manual de utilização do ENQUIRE está disponível em: http://infomesh.net/2001/enquire/manual/

[6] São coleções de documentos editados utilizando hipertexto, são caracterizados por ser editados coletivamente, são artigos que podem ser complementados com comentários de qualquer usuário da Internet, maiores informações em: http://pt.wikipedia.org/wiki/Wiki

Em 1989, Lee criou um artigo chamado "*Information Management: A Proposal*", que tratava de um documento mais formal, referenciando o sistema criado em 1980, o ENQUIRE. O artigo teve uma repercussão muito grande e circulou por todo o CERN.

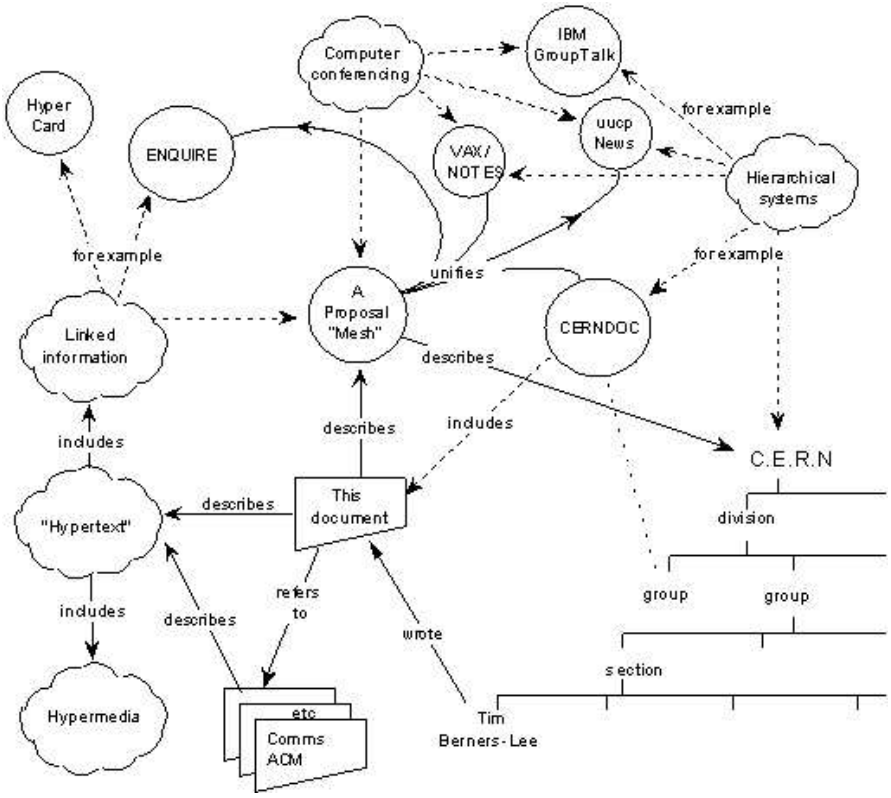


Figura 1-2 Proposta de Tim Berners-Lee

Fonte: http://www.w3.org/History/1989/Image1.gif

A Figura 1-2 apresenta uma representação gráfica da proposta desenvolvida por Berners-Lee. Alguns dos itens que descreviam os requisitos que deveriam ser atendidos pela solução são mostrados a seguir:

- Acesso remoto através de diferentes redes;
- Compatibilidade entre sistemas diferentes, na época chamada de heterogeneidade, devido a presença de computadores que rodavam sobre diferentes plataformas;
- Sem centralização, permitindo a criação de novos pontos onde fossem necessários;
- Acesso a dados existentes;
- Favoritos, na época chamado de "Private Links".

A proposta de 1989 apresenta uma possível solução para viabilizar a interconexão dos computadores, através de uma camada capaz de abstrair as diferenças das plataformas, pois esta interceptaria as requisições e faria o acesso as informações. A figura 1-3 mostra a arquitetura proposta para viabilizar a comunicação entre computadores de diferentes plataformas.

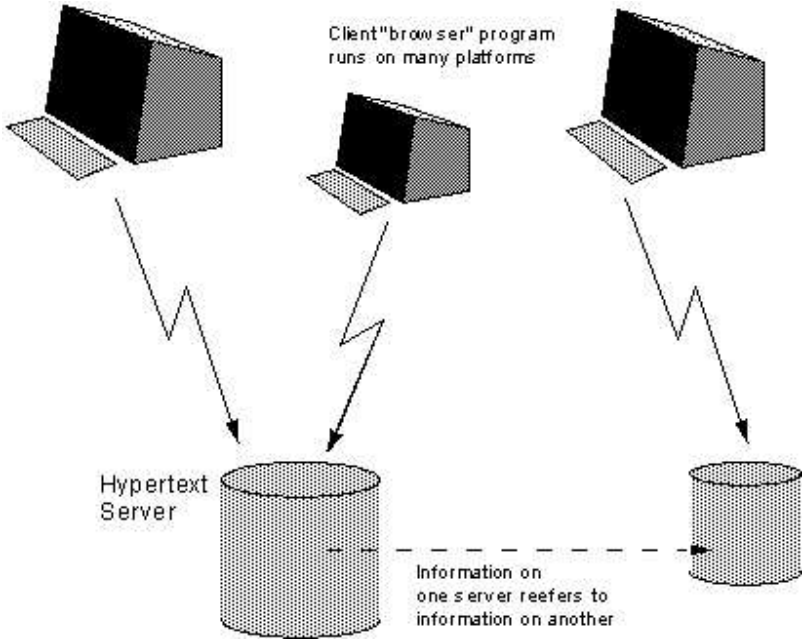


Figura 1-3 Comunicação entre diferentes plataformas

Fonte: http://www.w3.org/History/1989/Image2.gif

Em 1989, o CERN era o maior nodo da Internet na Europa, o que despertou o interesse de Tim, pois um dos problemas, o geográfico, poderia ser contornado utilizando-se a Internet. Foi nesse cenário que surgia a oportunidade única para Berners-Lee viabilizar o seu projeto, juntando os seus estudos sobre o sistema baseado no hipertexto com a Internet, [...] "Eu apenas precisava pegar a idéia do hipertexto e conectar com as idéias do TCP e DNS e juntando tudo, surgiu a World Wide Web."[...] (WIKIPEDIA, 2006c) (Tradução nossa).

Nessa época, sob uma plataforma NEXT, criou o primeiro servidor Web batizado de httpd "*HyperText Protocol Transfer Daemon*" e o primeiro navegador, batizado de WorldWideWeb, mais tarde o rebatizou para Nexus, a fim de evitar as confusões entre o serviço e o navegador.

O navegador proposto por Berners-Lee, era revolucionário, permitia navegar entre documentos, referenciados entre si e já vinha com um editor embutido.

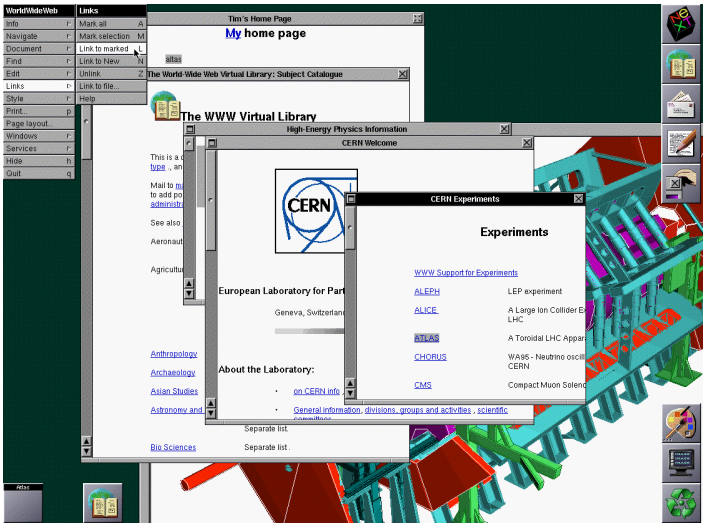


Figura 1-4 Tela do navegador WorldWideWeb

Fonte: http://en.wikipedia.org/wiki/WorldWideWeb

O pesquisador tentou implantar sua recém criada invenção no CERN, para oferecer uma maneira de ligar os documentos entre os diferentes e incompatíveis sistemas utilizados. Após algum tempo disponibilizou um pacote, tornando o uso do WorldWideWeb livre para outros centros de pesquisa e universidades, esse era composto por:

- Navegador (WorldWideWeb);
- Servidor Web (httpd);
- Uma biblioteca com implementações básicas de funções permitindo que desenvolvedores criassem seu próprio software;

Assim, com o crescimento da utilização da Web, novas necessidades surgiram que foram rapidamente difundidas no meio acadêmico, incentivando vários estudantes a trabalhar e propor soluções para o desenvolvimento de um navegador que apresenta-se maior compatibilidade entre as diferentes plataformas. Como o WorldWideWeb havia sido desenvolvido sob uma plataforma específica, a NEXT, era necessário que interfaces que viabilizassem o conceito *point and click* (aponte e clique) para outras plataformas como: PC, Unix, Mac, entre outros.

Os estudos realizados no CERN logo se espalharam para o mundo todo, e em 1991, novamente em um centro de pesquisa, o SLAC (*Stanford Linear Accelerator Center*), na

Califórnia, entra em funcionamento o primeiro servidor Web dos EUA. Nessa época, o cenário era caracterizado por dois tipos de navegadores:

- O original, proposto por Berners-Lee, bastante sofisticado, mas rodando apenas em computadores da plataforma NEXT;
- E "os outros", conhecidos como navegadores "*line-mode*", fáceis de instalar em diferentes plataformas, mas muito limitados nas funcionalidades e na interface com o usuário.

Com a popularidade no mundo acadêmico, surgiram alguns projetos individuais, sem muito sucesso, dentre eles pode-se destacar:

- MIDAS
- Viola
- Erwise

Os projetos individuais continuaram, até que em 1993, surge o MOSAIC, no NCSA, desenvolvido sob um ambiente *X Window*, muito conhecido na comunidade, permitindo uma interface com o usuário baseada em janelas. Esse navegador já apresentava os primeiros sinais de portabilidade entre plataformas, foi desenvolvido para as plataformas PC e Mac. O sucesso alcançado com o MOSAIC levou seu criador, Marc Andreesen a fundar a *Mosaic Communications Corporation*, que mais tarde originou a empresa *Netscape*.

Segundo Zakon, (2005, p.11), o crescimento era notório, o tráfego de informações gerado pela Web crescia 341,634% anualmente.

1.2.3 A evolução da World Wide Web

A competitividade entre as empresas que buscavam criar soluções atrativas para a Web estava aumentando. Com o objetivo de não permitir que essa competitividade gerasse um meio poluído, Tim saiu do CERN e mudou-se para o laboratório de Ciências da Computação do MIT para fundar o W3C em 1994.

O principal objetivo era criar uma aliança entre entidades comerciais, educacionais e governamentais que teria como objetivo levar a Web ao seu potencial máximo. Seguindo uma política aberta onde todos os membros poderiam participar de qualquer trabalho, projeto ou reunião, podendo opinar e direcionar os estudos para um determinado fim. Na sua criação já contava com o apoio do DARPA (*Defense Advanced Research Project Agency*) e da Comissão Européia.

Desde então o W3C se tornou uma entidade importante para uma regulamentação e padronização do desenvolvimento Web. É uma associação internacional de empresas que trabalham com o objetivo de desenvolver padrões Web. Com a seguinte missão: "Levar a World Wide Web para o seu maior potencial desenvolvendo protocolos e regras[...]" (W3C, 2006a,p.1) (Tradução nossa). Desde sua fundação em 1994, já publicou mais de noventa padrões, chamados de *W3C Recommendations*.

As principais iniciativas foram as ações visando estabelecer regras e padrões documentados, que poderiam ser seguidos como referência para o desenvolvimento de aplicações e troca de informações utilizando a Web, dentre elas, destacam-se:

- HTML – linguagem de marcação utilizada para "codificar" documentos de hipertexto. Além de serem utilizados na Web, são frequentemente utilizados para arquivos de ajuda;

- CSS – mecanismo que viabiliza a formatação de documentos de hipertexto. Possui sintaxe específica para estabelecer atributos (fonte, cor, cor do fundo, borda, preenchimentos) para cada elemento de um documento hipertexto;

- DOM – é uma plataforma que permite que scripts alterem a estrutura e o conteúdo de um documento dinamicamente, todo objeto de um documento HTML ou XHTML pode ser acessado utilizando e percorrendo a DOM;

- XHTML – linguagem de marcação com a mesma flexibilidade e utilização do HTML, porém baseada em XML, o que permite uma validação e padronização no seu formato, que passa a ser processado semanticamente;

- XML – tipo de documento concebido inicialmente para publicação eletrônica de conteúdos gigantescos, mas vem sendo utilizado de maneira satisfatória na troca de qualquer tipo de informação na Web;

- Internacionalização – conjunto de definições que permite a exibição de caracteres estendidos, viabilizando a exibição de conteúdo para qualquer idioma;

- WebServices – fornece padrões que viabilizam a troca de informações e integração de softwares rodando em diferentes plataformas, é tido como uma nova geração de aplicativos, também chamado de SOA, *Service Oriented Architecture*.

Muito do que se tem hoje em relação a padronização de aplicações Web, é devido ao esforço do W3C de propor padrões e funcionalidades que viabilizem o acesso à Web para todos e a partir de qualquer dispositivo, facilitando o acesso a informações de qualquer parte do mundo.

1.3 Aplicações Web

Durante muito tempo a Web foi encarada como um meio de trocar informações e documentos em formato hipertexto, mas a capacidade de acesso a partir de qualquer lugar no mundo e qualquer que seja a plataforma, despertou o interesse de muitos no que tange também ao desenvolvimento de aplicativos.

A troca de mensagens nestes aplicativos, entre o cliente e o servidor, normalmente é um processo lento, pois a cada requisição do cliente é esperado o retorno do servidor para que então a próxima requisição seja realizada. Este mecanismo de troca de mensagens é conhecido como conexão síncrona.

Não menos importante, a ausência de um mecanismo que permita armazenar o estado da aplicação no servidor cria problemas no âmbito da duplicidade de dados que devem ser trafegados para cada requisição, isso acontece devido ao fato das requisições sempre partirem do cliente para o servidor, jamais ao contrário, pois o servidor apenas responde a requisição do cliente, não consegue identificá-lo para fazer uma chamada.

Essas limitações existem e, durante muito tempo, foram contornadas com a utilização de técnicas alternativas que permitissem simular a troca de mensagem assíncrona, uma das mais conhecidas é a utilização de FRAMES ou IFRAMES. Essa solução consiste em direcionar a chamada a um elemento oculto encarregado de repassar a requisição para o servidor. Sendo assim, a aplicação principal, ou, nesse caso o frame principal, não fica "ocupado", permitindo que a navegação flua naturalmente e ao receber a resposta o elemento oculto informa o elemento principal e o resultado é mostrado. Essa solução funciona, é bastante difundida, porém oferece alguns pontos negativos:

- Gera um *overhead*[7] na página web desnecessário com objetos ocultos, responsáveis pela comunicação;
- Não é compatível com os padrões estabelecidos pelo W3C, ou seja, não passaria nos processos de validação semântica;

Devido a esses problemas soluções alternativas tiveram que ser criadas. É nesse contexto que surge o objeto *XMLHTTPRequest*.

1.3.1 XMLHttpRequest

Segundo Wikipedia (2006,p.1), o *XMLHttpRequest* é uma API que permite transferir dados em formato XML de e para servidores Web, utilizando o protocolo HTTP como meio de transferência. Esta API é acessível através de JavaScript, VBScript e outras linguagens de *script* suportadas pelos navegadores.

Foi implementado originalmente no navegador da Microsoft, o Internet Explorer 5.0, com o nome XMLHTTP, através de um controle ActiveX[8]. Em 2002, os navegadores da fundação Mozilla, implementaram suporte nativo ao XMLHTTPRequest, essa implementação foi seguida pelos demais navegadores, como é retratado por Wikipedia (2006,p.2):

- Safari 1.2;

[7] Usado na computação para indicar o excesso ou o valor agregado sobre um recurso: tempo, memória, largura de banda para realizar determinada tarefa.
[8] É uma tecnologia criada pela Microsoft para facilitar a integração entre aplicações.

- Konqueror;
- Opera 8.0;

Conforme W3C (2006c, p.1):

> O objeto XMLHttpRequest é implementado atualmente, de alguma forma, pelos mais populares navegadores Web. Infelizmente as implementações não são completamente interoperáveis. O objetivo dessa especificação é para documentar o mínimo de funcionalidade interoperáveis baseada em implementações existentes, permitindo que desenvolvedores Web usem essas funcionalidades sem a necessidade de código específico da plataforma. Sendo assim, somente funcionalidades que já estão implementadas são consideradas.[...] (Tradução nossa).

A popularidade no uso do XMLHttpRequest, levou o W3C a criar uma especificação, chamada *Document Object Model (DOM) Level 3 Load and Save*, que visa criar meios para a obtenção de um nível mínimo de interoperabilidade.

1.3.2 As origens do AJAX

Acrônimo para *Asynchronous Javascript And XML*, sem dúvida um dos termos mais falados na atualidade, vem revolucionando o desenvolvimento de aplicações Web, permitindo uma melhor interação com o usuário.

Conforme Wikipedia (2006c), o objetivo é fazer com que páginas Web se tornem mais interativas, através da troca de dados com o servidor, sem a necessidade do congelamento e das longas esperas pelo retorno do mesmo, sendo assim, não há necessidade de recarregar a página inteira, toda vez que o usuário realiza uma alteração.

O termo AJAX foi cunhado por Jesse James Garrett, em 2005, no artigo *AJAX: A New Aproach to Web Applications*, disponível em http://www.adaptivepath.com/publications/essays/archives/000385.php .

> [...] é um novo termo para duas poderosas funcionalidades de um navegador, que já existiam a anos, mas despercebida para vários desenvolvedores Web até recentemente, quando aplicações como Gmail, Google Suggest e Google Maps chegaram[...] (MOZILLA (2006, p.1) (Tradução nossa)

As duas funcionalidades são:

- Fazer requisições para o servidor sem recarregar a página;
- Trabalhar e tratar documentos XML.

AJAX, não é uma nova tecnologia, mas sim um conjunto de tecnologias presentes no dia a dia do desenvolvimento Web que foram agrupadas e atuam cooperando entre si, buscando um fim específico. Garrett, (2005,p.1), retrata as tecnologias envolvidas:

- Visualização baseada em *Web Standards*[9],utilizando XHTML e CSS;
- Visualização dinâmica e interativa usando o *Document Object Model* (DOM);
- Troca e manipulação de dados através de XML e XSLT;
- Obtenção de dados de forma assíncrona, utilizando o XMLHttpRequest;
- JavaScript, para tornar possível a junção de todos os anteriores.

Conforme Steil e Camargo (2005), embora muitas pessoas consideram que não haja motivos para tanta aclamação ao AJAX, ele é um dos principais responsáveis pelo surgimento da Web como plataforma.

O objetivo deste capítulo foi apresentar a evolução e principais marcos que contribuíram para que a Web chegasse no seu estado atual, indo muito além da proposta inicial, que era viabilizar a troca de documentos de hipertexto. No próximo capítulo serão apresentados maiores detalhes da tecnologia AJAX, através de exemplos práticos, mostrando as vantagens do uso da mesma no desenvolvimento de aplicações Web.

[9] Conjunto de padrões definidos pelo W3C que visam definir modelos das páginas que passam a ser consideradas semanticamente corretas.

2 AJAX

Até o momento, foi apresentado um resumo dos principais marcos da história da Web, foi possível perceber a clara evolução que ocorreu, até se chegar ao modelo mais avançado das aplicações Web, também chamadas de aplicações RIA[10], acrônimo para *Rich Internet Applications*, aplicações ricas na Internet.

O uso da Web durante muito tempo foi limitado à publicação de conteúdo em documentos de formato hipertexto, onde o HTML e suas variações em combinação com o CSS formavam uma dupla que atendia satisfatoriamente as necessidades para o fim que foram concebidos. A possibilidade de criar aplicações sob a plataforma Web despertava o interesse de muitas pessoas, as tecnologias do lado do servidor (*server side*) evoluíam e possibilitavam essa nova necessidade, porém as funcionalidades básicas para uma interação entre usuário e aplicação não eram oferecidas pelos navegadores, que apresentavam tecnologias muito limitadas, trazendo a necessidade de usar maneiras alternativas para se obter uma interface um pouco mais rica, o esforço não compensava. É nesse cenário que o uso de AJAX agrega valor ao desenvolvimento Web, permitindo uma interação que até então não era possível, sem perder a leveza e os benefícios do uso de um documento HTML.

O objetivo deste capítulo é apresentar as vantagens e considerações no uso da tecnologia AJAX para o desenvolvimento de aplicações Web ricas.

---

[10] *Rich Internet Applications*, nova geração de aplicações Web, caracterizadas principalmente pela qualidade na interface com o usuário. Maiores informações podem ser obtidas em: http://www.adobe.com/resources/business/rich_internet_apps/

2.1 AJAX em detalhes

Conforme citado anteriormente, AJAX é uma API, formada por uma combinação de tecnologias que foram utilizadas durante muito tempo e conviveram despercebidamente com muitos profissionais da Web.

Segundo Garrett(2005,p.1), a interação de um usuário com uma aplicação Web tradicional, durante muito tempo se deu da seguinte forma:

Figura 2-1 Arquitetura Clássica de Aplicações Web

Imagem do autor, adaptado de GARRETT (2005)

A Figura 2-1 apresenta o modelo de arquitetura padrão de uma aplicação Web, onde o navegador dispara uma requisição HTTP, o servidor Web, acessa os sistemas legados, busca os dados, e devolve as informações para o navegador através de HTML e CSS gerados. Então, a reposta é mostrada ao usuário.

O modelo exemplificado, e chamado de modelo tradicional, inviabilizava totalmente o desenvolvimento de uma aplicação mais complexa, uma vez que, enquanto o servidor realizava o processamento, a interface com o usuário permanecia congelada e indisponível até o recebimento da resposta e o recarregamento de toda a página.

Esse é um dos problemas que podem ser solucionados através da utilização de AJAX nas aplicações Web, através do suporte a chamadas assíncronas ao servidor, conforme será abordado na próxima seção. McLellan (2005, p.1) exemplifica:

> "Considere o exemplo simples de preencher o número de série de um produto em uma aplicação *desktop* em uma plataforma como o Microsoft Windows. Por convenção, assim que você finaliza a digitação da *string* alfanumérica, um ícone grande verde aparece, indicando que o código que você digitou é válido. Isso acontece instantaneamente devido a interface e a aplicação estarem interagindo, tão logo que você termina a digitação do número, a aplicação pode checar a validade e responder." (Tradução nossa)

No mesmo artigo, McLellan, faz a analogia a uma funcionalidade parecida, mas em uma interface Web, a interface com o usuário será, se não idêntica, muito similar a aplicação *desktop*. Porém, ao completar a digitação, o usuário terá que enviar (dar *submit*) a página para o servidor, com o objetivo de validar o conteúdo da informação digitada, sendo assim uma nova página será carregada com uma mensagem, informando o sucesso ou não, e no caso de não sucesso o usuário deverá voltar a página anterior, e digitar novamente a informação até que a mesma esteja correta.

Uma solução que parece óbvia seria colocar a validação no cliente. Assim, a nova requisição ao servidor com o objetivo de validar a informação seria evitada, porém, nesse caso tem-se uma validação simples, mas e em casos onde a validação depende de inúmeras verificações, aumentando a complexidade? Conforme McLellan, a interface com o usuário, não é o local correto para validações complexas.

Considerando a mesma arquitetura mostrada anteriormente, a figura 2-2 apresenta como ficaria a arquitetura da mesma aplicação utilizando AJAX:

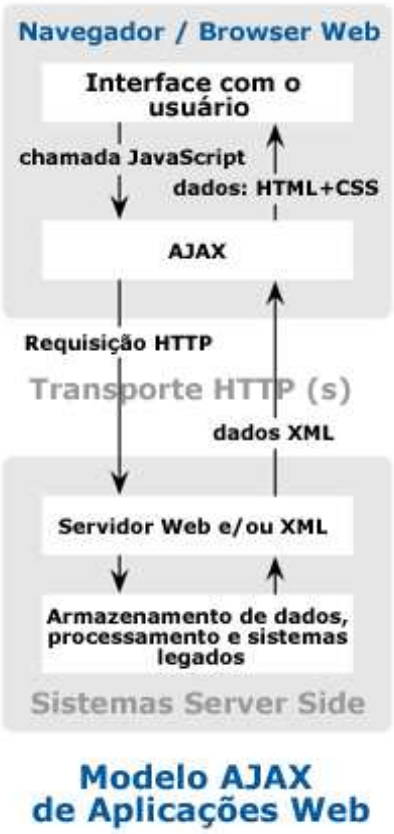


Figura 2-2 Arquitetura de Aplicações Web utilizando AJAX

Imagem do autor, adaptado de GARRETT (2005)

Como é possível perceber, quem passa a fazer a requisição para o servidor Web, não é mais a aplicação em si, mas ela passa a ser feita com o auxílio dos componentes que formam o AJAX. A página não fica mais congelada enquanto o servidor realiza o processamento da requisição, e a resposta é recebida de maneira semelhante a arquitetura tradicional.

2.1.1 Como tudo funciona?

A chave para a utilização do AJAX em uma aplicação Web, é o uso do objeto XMLHttpRequest, que deve ser instanciado e configurado. Ele dispõe de alguns atributos que permite ajustar o comportamento da requisição.

Primeiramente precisa-se instanciar[11] um objeto XMLHttpRequest. Existem maneiras diferentes de se fazer isso nos navegadores atuais, visto que, o Internet Explorer não

[11] Termo utilizado na programação orientada a objetos, é o ato de criar uma instância de um objeto ou classe.

suporta nativamente este objeto, e é implementado através de um componente ActiveX. Abaixo se tem um exemplo de como fazê-lo:

Internet Explorer:

```
var req = new ActiveXObject("Microsoft.XMLHTTP");
```

Firefox / Opera / Safari:

```
var req = new XMLHttpRequest();
```

Uma solução alternativa é utilizar um método que verifique a existência do objeto nativamente no JavaScript, caso contrário instanciar o objeto ActiveX:

```
function getXMLHTTPRequest() {
    var xRequest=null;
    if (window.XMLHttpRequest) {
        xRequest=new XMLHttpRequest();
    }else if (typeof ActiveXObject != "undefined"){
        xRequest=new ActiveXObject("Microsoft.XMLHTTP");
    }
    return xRequest;
}
```

O código acima, verifica a existência do suporte nativo ao objeto XMLHttpRequest e instancia o objeto de acordo com o resultado do teste. De posse do objeto instanciado corretamente é necessário fazer as configurações da requisição:

```
var req = getXMLHTTPRequest();
req.open("POST","http://www.servidor.com.br/arquivo.php");
req.onreadystatechange=funcao_handler;
req.send("nome=pedro&idade=25");
```

Para o exemplo acima, obteve-se um objeto do tipo XMLHttpRequest, utilizando a função `getXMLHttpRequest()` (que foi implementada anteriormente, para abstrair as diferenças entre as plataformas Internet Explorer X outros navegadores), foi definida uma requisição que utilizará o método POST do protocolo HTTP e realizará uma chamada para a URL http://www.servidor.com.br/arquivo.php, a URL de destino receberá duas variáveis. São elas: nome, como o valor pedro e idade com o valor 25.

Outro ponto que merece destaque no uso do AJAX é a definição do ponto de reencontro, ou para onde vai o fluxo, após o recebimento da resposta da requisição? Através do parâmetro `onreadystatechange` é possível definir o nome de uma função que será chamada a cada alteração do status da requisição. Para o exemplo anteriormente mostrado, tal função é a `funcao_handler`, definida da seguinte forma:

```
function funcao_handler (){
      if(req.readyState==4)    {
            alert("Retorno do script php: "+ req.responseText);
            req.close();
      }
}
```

O estado da requisição pode ser obtido através do atributo `readystate`. Na implementação da função funcao_handler, existe o teste verificando se o valor do atributo é 4, caso verdadeiro, o retorno da requisição é mostrado com a função alert(), que irá exibir uma caixa de mensagem para o usuário.

A Tabela 1 apresenta os possíveis valores para o atributo readystate:

Tabela 2-1 Valores do atributo readyState

Valor	Estado da requisição
0	Requisição não inicializada
1	Requisição sendo feita
2	Requisição feita
3	Requisição interagindo
4	Requisição Concluída

O exemplo apresentado é bastante simples, sendo que foi criado com o objetivo de demonstrar o funcionamento e o ciclo de vida de uma requisição AJAX. A implementação da função `funcao_handler`, ou qualquer outra que tenha sido definida no atributo `onreadystatechange` do objeto criado, poderia implementar qualquer lógica, como por exemplo:

- Tratar os dados recebidos e inserir linhas ou colunas em uma tabela;
- Atualizar o estado de qualquer componente que estiver na tela: ativar e desativar um botão;
- Atualizar o valor ou conteúdo de qualquer componente que estiver na tela, um campo de totalização;

Essa flexibilidade de manipulação em qualquer componente, objeto, texto, enfim, todos os itens que formam a interface com o usuário é devido ao suporte as especificações do W3C relacionadas a DOM e implementadas pelos navegadores Web.

É importante destacar que toda e qualquer operação realizada por um script *server side*[12], pode ser chamado utilizando AJAX. Por exemplo, na seguinte situação: após a perda de foco de uma caixa de texto que deveria conter um endereço de e-mail válido, a página Web dispara uma requisição assíncrona para um script no servidor responsável pela validação da informação digitada, com o objetivo de testar a validade do e-mail informado.

A representação do input utilizando HTML, poderia ser assim:

```
<input    type="text"
          name="eMail"
          id="eMail"
          size="50"
          onBlur="validaEmail();" />
```

A função JavaScript `validaEmail` poderia ter a seguinte implementação:

```
var myReq;
function validaEmail(){
      var objInputEmail = document.getElementById("eMail");
      if(objInputEmail.value != ""){
            myReq = getXMLHTTPRequest();
            myReq.open("POST", "valida_endereco_email.php");
            myReq.onreadystatechange=retornoValidacao;
            myReq.send("email=" + objInputEmail.value);
      }
```

[12] Scripts que não executam no navegador / cliente, rodam no servidor, exemplos: ASP, PHP, Coldfusion.

```
}
```

Apenas para revisar, a função `validaEmail` está testando o valor digitado no campo `eMail` do formulário. Caso ele tenha algum valor, é criado um objeto XMLHttpRequest, configura os atributos do objeto para satisfazer a necessidade da requisição, nesse caso será feita uma requisição utilizando o método POST para o arquivo valida_endereço_email.php, passando uma variável chamada email, contendo o valor digitado no campo. Após a requisição mudar de status, a função `retornoValidacao` será chamada. Uma possível implementação para a função seria:

```
function retornoValidacao (){
      if(myReq.readyState==4){
            if(myReq.responseText != "ok"){
                  alert("O e-mail informado é inválido, por favor informe
um e-mail válido.");
            myReq.close();
      }
}
```

Ao invés de ter uma validação apenas no cliente, poderia ser implementada uma validação que tentaria conectar no servidor de email, do endereço fornecido e verificar a validade do mesmo. No caso de ser um email válido o script do servidor retornaria a string ok, que indicaria a validade da informação fornecida.

2.2 Quando utilizar

Como toda tecnologia, o uso abusivo de AJAX pode prejudicar o desenvolvimento de uma aplicação, funcionalidades muito simples, não justificam serem implementadas utilizando a tecnologia. Com esse objetivo essa seção apresentará situações onde o uso de AJAX é necessário e aconselhável.

Segundo Murray (2005, p.1) as seguintes situações são plausíveis do uso de AJAX:

- Validação de dados de formulário em tempo real – Informações de um formulário, por exemplo: códigos de identificação, números de série, CEPs, são dados que precisam de validação do servidor, antes mesmo do usuário submeter o formulário;

- Dicas / Sugestões (recurso de *Autocomplete*) – Situações onde, ao digitar apenas as iniciais de um dado, é possível sugerir possíveis valores para determinado campo;

- Operações Mestre-Detalhe – A partir de um evento ocorrido no cliente, uma página HTML pode buscar maiores informações sobre um determinado item selecionado. Por exemplo a busca de informações de um produto, ao clicar em um item de uma lista;

- Componentes avançados da interface com o usuário – Componentes como, *tree controls*, menus e barras de progressos podem ser manipulados sem a necessidade de recarregar a página;

- Atualizar dados em uma pagina – Quando for necessário dados atualizados do servidor;

- Notificações do servidor – Casos em que o servidor precisa informar ao cliente sobre determinada situação, através da exibição de uma mensagem, atualização de uma informação ou redirecionamento de página, por exemplo, o término de *upload* de um arquivo que estava sendo realizado em *background*[13].

2.3 Benefícios

Até o momento o trabalho focou em aspectos técnicos do uso do AJAX, o objetivo dessa seção é trazer os benefícios de um ponto de vista administrativo, auxiliando os técnicos a convencer seus gestores a aderir à tecnologia.

Segundo Jensen (2006,p.2), as diferenças entre os aspectos técnicos e gerencias, podem ser definidos da seguinte forma:

Tabela 2-2 Diferenças entre os aspectos técnicos e gerenciais

Desenvolvedores falam sobre	Gestores falam sobre

[13] Processo que roda paralelamente ao fluxo normal da aplicação, permanece executando sem interromper a aplicação principal.

Usabilidade	Eficiência
Largura de banda	Custo de transações
Ramp-up[14]	Custos com treinamento
TCO[15]	ROI (Retorno sobre o investimento)

Fonte: Do autor, adaptado de Jensen

Medir os benefícios de um aplicação Web que utiliza AJAX é uma tarefa complicada, pois não há uma métrica definida para estabelecer os benefícios em aspectos gerenciais de uma melhor interface para o usuário.

Conforme White (2006,p.2), os benefícios podem ser extraídos através de alguns indicadores que podem ser mensurados, são eles: o tamanho de banda utilizado, e o tempo necessário para realizar determinada tarefa. Através de um caso representativo, foi possível estabelecer que em um processo constituído de 10 etapas é possível economizar de 500 a 2800 horas de trabalho por ano, economizando 4 segundos por etapa.

Figura 2-3 Comparativo do tempo consumido

Imagem do autor, adaptado de Jensen(2006,p.3)

[14] Capacidade de aumentar a produtividade de uma empresa, antecipando-se a uma possível demanda, agir antecipadamente, ação pró-ativa.
[15] Índice financeiro para mensurar custos diretos e indiretos na compra de algo.

Como é mostrada na Figura 2-3, a implementação utilizando a arquitetura AJAX permitiu que o processo escolhido fosse executado em menos tempo que a arquitetura tradicional.

Figura 2-4 Comparativo de bytes transferidos

Imagem do autor, adaptado de Jensen(2006,p.3)

A implementação utilizando a arquitetura AJAX transfere menos dados para realizar o procedimento hipotético definido no experimento.

2.4 Desvantagens

Mesmo trazendo benefícios enormes para aplicações baseadas em AJAX, alguns pontos negativos e novos desafios devem ser encarados, principalmente no que tange a maneira como essas são desenvolvidas. Conforme Telerik (2005,p.6), alguns dos desafios a serem vencidos são:

- Escrever e manter scripts complexos desenvolvidos utilizando JavaScript;
- Necessidade de manter o estado de uma tela;
- Mudança na maneira como o desenvolvimento de web sites ocorre, uma vez que o conceito de página, não é mais válido;

- As novas interfaces com o usuário, mais interativas trazem novos problemas a serem tratados, por terem um funcionamento não esperado e não habitual para os usuários.

Além disso, a navegabilidade da aplicação muda, uma vez que o conteúdo não é recarregado, o botão voltar do navegador não vai detectar a mudança de uma página, sendo assim, não será possível retornar à pagina anterior a partir da atual. Isso ocorre devido ao fato do conteúdo ser carregado dinamicamente, contrariando a maneira padrão de funcionamento. Além do problema do não funcionamento do botão voltar, não será possível acessar determinada pagina que teve seu conteúdo alterado, através de uma URL direta, isso implica em não poder inserir a pagina atual, com o conteúdo atual nos favoritos, para uma visita futura.

Devido a esses problemas, algumas soluções já foram desenvolvidas, mas são funções auxiliares utilizadas com o objetivo de contornar o problema que o uso de AJAX traz.

2.5 Frameworks

O surgimento do AJAX e a popularidade que vem ganhando atualmente trouxeram como conseqüência o surgimento de muitas ferramentas que auxiliam os desenvolvedores, facilitando tarefas complexas, através de abstrações que tornam o uso de AJAX mais fácil, sem a necessidade de complexas implementações, facilitando o ajuste / *refactoring*[16] de aplicações existentes.

Essa seção fará um apanhado geral dos principais frameworks que vem ganhando espaço e sendo utilizados, trazendo novas funcionalidades além das propostas já vistas até o momento. Os frameworks podem ser classificados em dois grandes grupos:

[16] Denominação para uma técnica da engenharia de software que permite alterar o código fonte de algum módulo sem alterar o seu comportamento ou a interação do mesmo com demais módulos.

- Os desenvolvidos com o objetivo de facilitar a integração com alguma tecnologia *server side*[17], por exemplo o DWR, que permite que objetos JAVA sejam instanciados via JavaScript;

- Os que visam facilitar o acesso a DOM e ao objeto XMLHttpRequest, através de abstrações que permitem que o trabalho braçal fique todo em um determinado método.

Os principais frameworks são mostrados a seguir e descritos em linhas gerais:

- DOJO – É um conjunto de bibliotecas JavaScript que oferecem uma API simples com uma série de funcionalidades, dentre elas, funcionalidades para trabalhar com AJAX. Outras funcionalidades como: manipulação de strings, manipulação da DOM, *drag-and-drop*, estruturas de dados como listas, filas e pilhas. http://today.java.net/pub/a/today/2006/04/27/building-ajax-with-dojo-and-json.html

- Prototype – É um framework JavaScript que viabiliza o desenvolvimento de aplicações Web dinâmicas, permitindo o desenvolvimento em classes e oferecendo a melhor implementação para se trabalhar com AJAX http://prototype.conio.net/;

- DWR – Projeto que permite instanciar objetos JAVA que são expostos e disponibilizados para interagirem com JavaScript;

- SAJAX – É um projeto de código aberto que viabiliza a chamada a scripts alojados no servidor codificados nas linguagens PHP, Perl ou Phynthon.

2.6 Casos de sucesso

A popularidade do AJAX e muito do que se tem mostrado são oriundos de empresas consagradas no mundo da tecnologia de informação, principalmente no mundo da Internet,. Empresas como Google, Yahoo, Microsoft, IBM vêm investindo pesado e apresentando soluções que ganham espaço e conquistam usuários em todo o mundo. A grande credibilidade

[17] Script ou conjunto de instruções que são processadas no servidor e o resultado do processamento pode ser usado para gerar saída para o cliente.

que AJAX vem ganhando no mundo atual, pode ser atribuída à quantidade de grandes empresas que vem colaborando para o desenvolvimento, padronização e divulgação da mesma.

Recentemente a IBM junto com outras 14 empresas que atuam no ramo de tecnologia de informação anunciaram a criação da primeira comunidade *open source*, que promoverá o uso do AJAX. Além da IBM, farão parte dessa comunidade: BEA, Borland, Dojo Foundation, Eclipse Foundation, Google, Laszlo Systems, Mozilla Corporation, Novell, Openwave Systems, Oracle, Red Hat, Yahoo, Zend e Zimbra, conforme IBM (2006,p.1).

Outra notícia que chamou a atenção, foi o lançamento da *Google Web Toolkit*, pela empresa Google, que consiste em bibliotecas JAVA, que permitem criar aplicações Web baseadas em AJAX.

As vantagens oferecidas através da utilização de AJAX, sem dúvida agregam funcionalidades a uma aplicação Web, permitindo uma nova gama destas, que anteriormente só eram possíveis em aplicações *desktop*, ou através de *workarounds*[18], que acabavam complicando e exigindo muito esforço dos desenvolvedores, não justificando a utilização.

Conforme Murray (2005, p.1), qualquer usuário que tenha se deparado com aplicações Web como, Flicker, Gmail, Google Suggest ou Google Maps, percebe as inovações na dinamicidade que vem tomando conta das aplicações Web modernas.

O surgimento de aplicações Web capazes de substituir ferramentas utilizadas no cotidiano da maioria dos usuários, traz uma infinidade de benefícios para uma organização, dando mobilidade e praticidade para o usuário final, que passa a ter uma ferramenta utilizada no dia-a-dia disponível em qualquer lugar que tenha acesso a Internet. Isso se comprova através dos exemplos mostrados, Google Calendar, Gmail, Writely, Google SpreadSheets, del.icio.us, Flicker entre outras inúmeras ferramentas que estão substituindo aplicações desktop e trazendo a funcionalidade para um navegador Web.

[18] Solução que atinge os objetivos a que se propõe, porém, muitas vezes de forma não clara ou indo contra boas práticas.

3 RELATÓRIOS

Este capítulo contextualizará a situação dos atuais sistemas de informação, quanto ao desenvolvimento de relatórios. A partir da contextualização da necessidade de customização da ferramenta, serão apresentadas as dificuldades encontradas na maioria dos sistemas de informação atuais, no que tange a adaptabilidade de funcionalidades.

3.1 Importância

O meio corporativo, nos dias de hoje, caracterizado por empresas que lidam com uma infinidade muito grande de informações, traz a necessidade de organização, constante avaliação e medição de indicadores, permitindo análise estratégica e auxílio na tomada de decisões.

O aglomerado de informações gerado pelas empresas que atuam nos mais diferentes ramos de atuação trouxe a necessidade de técnicas específicas para o armazenamento e restauração da informação julgada necessária e relevante sobre um determinado contexto.

Para o correto entendimento do estudo, faz-se necessário apresentar a definição de informação, que foi definida por Barreto (apud Moresi, 2000, p.2) como: estruturas significantes com a competência de gerar conhecimento no indivíduo, em seu grupo, ou na sociedade. É possível complementar com a idéia de Bogui e Shitsuka (2002, p.34), onde informação é considerada a matéria-prima para a tomada de decisão.

Sendo assim, a disponibilidade de maneira rápida e precisa às informações, passa a ser de fundamental importância para o sucesso das organizações, podendo ser encarada como um poderoso recurso, que agrega valor ao produto ou serviço final, conseqüentemente oferecendo um diferencial competitivo.

Por apresentar uma estrutura diferente de outros recursos, não permitindo a categorização em termos econômicos, o recurso informação, exige da organização a adoção de sistemas que permitam a gestão das mesmas, conforme Moresi (2000).

A adoção de sistemas de informação por organizações que atuam em uma ampla gama de serviços, não é novidade e vem ocorrendo desde o início da década de 90.

> A importância da informação para as organizações é universalmente aceita, constituindo, senão o mais importante, pelo menos um dos recursos cuja gestão e aproveitamento estão diretamente relacionados com o sucesso desejado. (MORESI, 2000,p.1)

Conforme retratado por Moresi, uma organização é subdividida em níveis. Sendo esses, compostos por diferentes públicos, que conseqüentemente têm diferentes perfis e necessitam de informações que vão de encontro ao perfil. Sendo assim, a geração da informação deve adequar-se a necessidade dos mesmos.

Conforme Business Object (2006a), "A criação de relatórios corporativos é um dos processos do *Business Intelligence*[19](BI) que permite acessar, formatar e disponibilizar as informações de maneira estratégica e com segurança pela Web ou por outras aplicações.".

Conforme Meyer, Baber e Pfaffenberger (2000,p.353), os sistemas de informações gerenciais são sistemas que produzem relatórios com a finalidade de avaliar o cumprimento de metas. Apesar de desempenhar um papel importante nas organizações, apresentam desvantagens, pois geram relatórios predefinidos que podem não conter as informações necessárias.

[19] Pode ser traduzido como inteligência empresarial, refere-se aos processos para a coleta e análise de informações empresariais. Maiores informações podem ser obtidas em: http://pt.wikipedia.org/wiki/Business_Intelligence

Além dos sistemas de informação gerenciais, a utilização de sistemas ERP também vem sendo praticada por organizações, com o objetivo de informatizar operações do nível operacional (DALL'OGLIO, 2002).

Segundo Bervian (2000), os sistemas ERP, assim como, os sistemas de informações gerenciais também são construídos seguindo as melhores práticas, sendo essas, conforme Souza e Swicker (apud BERVIAN,2000), obtidas através da experiência acumulada por empresas fornecedoras em repetidos processos de implantação. Davenport (apud BERVIAN,2000), afirma que nos sistemas ERP, é o fornecedor que define o que é a melhor prática, e não o cliente.

Em alguns casos as informações são armazenadas, coletadas, tratadas, mas não são plausíveis de apreciação e análise, através de uma forma mais formal, por exemplo, um relatório. Isso ocorre devido as impossibilidades de customizar os sistemas que, conforme citado anteriormente, são criados seguindo as melhores práticas, portanto, gerando, em alguns casos, desencontros com as necessidades e processos de uma organização.

3.2 Problemas encontrados atualmente no processo de geração de relatórios

Como citado anteriormente, alguns dos sistemas de informação tradicionais, apresentam limitações no que tange a personalização das necessidades dos clientes. Isso ocorre, pois os mesmos são concebidos utilizando como base as melhores práticas do mercado. Porém essa, nem sempre atende a empresa na integra, pois cada empresa possui seus próprios processos e maneira própria de trabalhar. Qualquer necessidade que fuja dos moldes do propósito pelo qual a ferramenta foi construída, gera problemas. A informação não estará disponível no momento que deveria. Uma decisão importante pode não ter sido tomada, ou tomada inadequadamente, devido a impossibilidade de análise das informações.

Conforme Dall'Oglio (2002), os sistemas são adaptados visando adequar-se à estrutura de processos e métodos da organização, assim como seus relatórios são modificados para que sua estrutura e conteúdos fiquem o mais próximo possível da realidade na qual o sistema esta sendo implantado.

Partindo do pressuposto que surge a necessidade de solicitação de adequação / personalização para o fornecedor do software, sejam eles, *software houses*, distribuidores do produto ou qualquer outra entidade, o pedido de personalização é feito e novamente espera-se o tempo de implementação do recurso, capaz de gerar a informação necessária. O recurso seja ele, uma funcionalidade ou relatório onera horas de trabalho do fornecedor do software, a organização acaba tendo custos adicionais que não estavam planejados.

Os relatórios costumam ser utilizados buscando atender a necessidade de diferentes públicos, sejam acionistas e diretores de empresas com informações gerenciais, que podem servir de embasamento para: o estabelecimento de metas, avaliação da empresa sob um determinado aspecto, entre outros. Ou ainda, o público externo, por exemplo, clientes buscando acompanhar o status de uma solicitação feita em um sistema de atendimento.

Essa dinamicidade, tanto de conteúdo, como de diferentes públicos, que geram a necessidade de estabelecer quais informações realmente são importantes para determinado fim, também pode gerar demanda de personalização e novas funcionalidades que devem ser agregadas ao software, onerando novamente o custo da empresa.

Essa dificuldade de personalização e adaptação dos sistemas de informações gerenciais é o objeto desse estudo, onde se pretende apresentar uma solução que viabilize a interação entre as informações e a criação e gerenciamento de relatórios.

3.3 Relatórios comumente utilizados

Atualmente é difícil imaginar qualquer aplicação que interaja com um SGBD que não ofereça ao menos uma operação de relatório. Conforme Castro (2006), os tipos de relatórios mais utilizados são:

- Listagem de dados: o mais conhecido tipo de relatório, pode ser uma listagem simples, ou dependendo do tipo de resposta a ser obtida, a ordernação e o agrupamento de dados pode dar forma ao relatório. Por exemplo, um relatório de pedidos por cidades;

- Gráficos: gráficos em geral;

- Cartas e documentos: documentos textuais que podem originar uma mala direta em combinação com o uso de registros de uma base de dados;

- Matrizes: exibir relatórios na forma de matrizes, com dados nos eixos X e Y.

- Relatórios compostos: seria um relatório formado por mais de um dos citados a-cima, por exemplo, um relatório de clientes com um gráfico de seus pedidos.

3.4 Ferramentas de relatórios

Para o completo entendimento do texto a seguir, se faz necessário a definição do termo ferramentas de relatório. Entende-se por ferramentas de relatório, aquelas que permitem a concepção / elaboração do relatório, não sendo consideradas ferramentas que obtém os dados e os aplicam sobre um determinado layout.

As tradicionais ferramentas de relatórios encontradas atualmente possuem características semelhantes: rodam em ambiente *desktop* e são concebidas sob a idéia de que serão utilizadas na etapa de planejamento e desenvolvimento do sistema, conseqüentemente utilizadas por profissionais da equipe de criação do produto, entre eles: projetistas, analistas e desenvolvedores. Freqüentemente disponibilizadas como API's, *plugins* ou extensões para a IDE, ficam fortemente dependentes do ambiente de desenvolvimento.

Existem consagrados fabricantes de ferramentas de relatório proprietárias, dentre eles: MicroStrategy[20], BusinessObjects[21], entre outras. A crescente evolução no uso de ferramentas de BI, conseqüentemente de *reporting*[22], também despertou a atenção da comunidade do software livre. Abaixo são listadas algumas soluções.

BIRT[23], criado pela Eclipse Foundation, significa *Business Intelligence and Reporting Tools*, fornecido em forma de plugin para a IDE Eclipse. "[...] Ele se propõe a desenvolver, com a qualidade dos demais projetos do Eclipse, a infra-estrutura e ferramentas para quem utiliza os conceitos de *Business Intelligence* (BI) [...]" (CASTRO,2006,p.10).

---

[20] Maiores informações em: http://www.microstrategy.com/
[21] Maiores informações em: http://www.businessobjects.com.br/
[22] Termo utilizado para identificar as tarefas e ferramentas de geração de relatórios.
[23] Maiores informações, bem como uma versão para *download* podem ser obtidas em http://www.eclipse.org

Conforme Business Object (2006b), o Crystal Reports, uma das pioneiras e mais utilizadas no mercado, encontra-se na versão XI. A ferramenta vem amadurecendo com o passar do tempo e aprimorando as funcionalidades e viabilizando agilidade na obtenção de informações no menor intervalo de tempo possível. Dentre as principais ferramentas, é possível destacar:

- Conectividade aos principais bancos de dados existentes no mercado;
- Interface amigável e auto-explicativa;
- Opções de agrupamento, inserção de fórmulas, totalizadores;
- Geração de relatórios a partir de uma interface Web;
- Geração de gráficos;
- Exportação dos relatórios em vários formatos;
- Opções de segurança, viabilizando o acesso diferenciado a informação baseado no perfil do usuário;
- Assistente para publicação de relatórios na Web;
- Agendamento de relatórios.

Além dessas funcionalidades, possui componentes que podem ser adicionados ao ambiente de desenvolvimento, facilitando a integração com a IDE. É possível integrar perfeitamente com tecnologias JAVA e .NET.

3.5 Nova geração das ferramentas de relatório

A deficiência e inexistência de ferramentas adequadas para auxiliar na tomada de decisão e na avaliação de resultados, trouxeram a necessidade do desenvolvimento de novas ferramentas, conforme é retratado por Lachev (2004, p.1):

> [...] De acordo com a Microsoft, os profissionais que trabalham com informação, gastam até 80% do tempo deles, coletando informações, tendo somente 20% do tempo para as analisar e tomar decisões. Em muitas empresas, essas tarefas

> consomem tempo significativo de recursos de desenvolvimento e de TI em geral. Freqüentemente, as ferramentas de relatórios predominantes nessas organizações são planilhas Excel, com a entrada de dados manual, esses são as principais causas da geração de dados incorretos e decisões tomadas de forma errônea. Sabendo disso a Microsoft iniciou o projeto Microsoft SQL Server 2000 Reporting Services [...] (Tradução nossa)

O uso do *SQL Server Reporting Services*[24] vem crescendo, e está se tornando uma ferramenta importante para auxiliar no processo de tomada de decisão das empresas.

[24] Maiores informações podem ser obtidas na edição nº 19 da Revista SQL Magazine.

4 A FERRAMENTA

De posse de informações sobre as melhores práticas do desenvolvimento de aplicações Web e após ter contextualizado o problema encontrado nos sistemas de informação utilizados atualmente, este tópico se propõe a apresentar a proposta de desenvolvimento, bem como a estrutura da ferramenta a ser concebida: um gerenciador de relatórios rodando sob a plataforma Web.

4.1 Descrição

A ferramenta, chamada de YARG[25], foi desenvolvida para suprir as necessidades de extração de informações de diferentes perfis de usuários, que por sua vez possuem diferentes objetivos. Essas informações, muitas vezes não estão disponíveis de maneira que viabilize o acesso simples e prático por usuários menos experientes e técnicos. O objetivo principal dessa ferramenta é possibilitar o uso da ferramenta por usuários não muito experientes.

Tal ferramenta permite a extração de informações, bem como a representação gráfica em forma de relatórios, distribuindo as informações de forma estruturada, com a finalidade de atender determinado objetivo. Para tal, será necessária, além da extração de informações de diferentes SGBD´s, a possibilidade de importação de fontes de dados alternativas, possibilitando assim, a exploração de dados oriundos de sistemas que não possam ser

[25] Acrônimo para *Yet Another Report Generator*, foi escolhido devido a um costume utilizado em comunidades *open source*, *Yet Another*, é frequentemente utilizado no nome dos projetos, estabelecendo uma espécie de padrão na nomenclatura.

acessados, seja por não utilizar um banco de dados baseado no padrão SQL, ou indisponibilidade de acesso a partir do ambiente Web.

Nesses casos, pretende-se através da exportação dessas informações em um arquivo XML, CSV, ou dados tabulares do sistema de origem, viabilizar o uso dessas informações pelo gerenciador de relatórios que importaria o mesmo e o trataria como uma fonte de dados SQL. Maiores detalhes sobre esse gerenciamento das fontes de dados são dados nas próximas seções desse capítulo.

4.2 Tecnologias utilizadas

Essa seção tem como finalidade apresentar as tecnologias utilizadas para a concepção da ferramenta.

4.2.1 PHP

A linguagem adotada para o desenvolvimento da aplicação, foi a PHP. Optou-se pelo uso dele, devido ao conhecimento avançado que o aluno e o orientador tinham da linguagem, além de que, na sua versão 5.0, é uma excelente ferramenta para o desenvolvimento de aplicações orientadas a objetos.

Conforme Meloni, 2004, é natural que uma linguagem fique em constante desenvolvimento e com a versão 5 do PHP, não foi diferente, as mudanças representam o próximo passo na evolução de qualquer linguagem de desenvolvimento.

O YARG foi concebido utilizando o paradigma de orientação a objetos sem nenhuma limitação para as tradicionais linguagens orientadas a objetos, JAVA, por exemplo.

4.2.2 XHTML e CSS

Como a ferramenta será concebida sob plataforma Web, faz-se necessário à escolha da tecnologia que será adotada para a concepção da interface com o usuário. As seguintes tecnologias foram avaliadas:

- JAVA *Applets* – Conforme SUN (2006, p.1), uma *applet* é um tipo de programa desenvolvido em Java, que é embutido em uma página Web, que pode rodar em um navegador, que suporte essa tecnologia. (Tradução nossa);

- XUL – Segundo XULPLANET(2006, p.1), XUL é uma linguagem baseada em XML que possibilita um desenvolvimento mais rápido e fácil para os navegadores da fundação Mozilla. Enquanto certas linguagens são caracterizadas pela complexidade altíssima do desenvolvimento de aplicações multiplataformas, o XUL foi especialmente desenvolvimento para a criação de interfaces portáveis. Outra vantagem é que a característica de não necessitar de compilação permite que uma interface em XUL, seja alterada muito rapidamente. (Tradução nossa);

- Macromedia Flash / Macromedia Flex / Laszlo interfaces baseadas em Flash – Web Application Solutions (2006, p.11), define o Flash, como um plugin que pode ser instalado na maioria dos navegadores, também pode ser usado com uma aplicação para desenvolver aplicações, apresentações, interfaces com o usuário. (Tradução nossa).

As opções citadas acima são excelentes alternativas para a concepção de uma interface rica com o usuário, porém todas elas apresentam limitações e certas condições a serem atendidas para utilizá-las. Algumas dessas limitações são apresentadas abaixo:

Tabela 4-1 Condições X Tecnologias

Tecnologia	Condição a ser atentida / Limitação
XUL	Roda apenas em navegadores da família Mozilla, sendo impossível, rodar no Internet Explorer, por exemplo.
Applet	As *applets* JAVA, precisam obrigatoriamente da maquina virtual JAVA instalada para serem executadas. Uma interface desse tipo, pode tornar-se lenta para uma aplicação baseada na Web (WEBAPPLICATIONS,2006).
Macromedia Flash / Flex / Laszlo	Necessita de plugin específico para a interpretação dos arquivos da interface.

Conforme a tabela 4-1, é possível constatar que ambas as tecnologias necessitam de algum software ou pré-requisito a ser atendido. Além da tabela acima, Web Application

Solutions (2006), ainda destaca algumas desvantagens na adoção de Java *Applets* e interfaces baseadas em Flash, dentre elas, cabe destacar:

- O conteúdo de ambas as tecnologias, não são indexados pelos serviços de busca, como o Google;
- Tem uma latência (tempo de espera), muito grande, devido à necessidade de baixar toda a interface, esse primeiro download, é frequentemente muito demorado e custoso para a infraestrutura;

Sendo assim, optou-se então, por utilizar XHTML e CSS, pois são linguagens de marcação baseadas em XML e HTML, permitindo um maior nível de padronização, visando atender a exibição em diferentes navegadores e dispositivos, como por exemplo: palm's, celulares, entre outros. Além disso, a adoção do XHTML, oferece uma alternativa leve, sem necessidades específicas de instalação e configuração de ambiente, uma vez que é especificado e mantido pelo W3C, que conforme Underwood, 2004, são os padrões consultados e utilizados pelos fabricantes dos navegadores.

Já o uso de CSS, além de trazer facilidade na alteração do layout, é utilizado com o objetivo de seguir a metodologia *tableless*[26], de modo a oferecer páginas Web mais leves e semanticamente corretas.

4.2.3 AJAX e JavaScript

Além da simplicidade e compatibilidade entre diferentes plataformas e ambientes, é necessário oferecer uma interface que permita uma interação rápida e intuitiva, para se alcançar essa característica, foi adotado o uso de AJAX, que permitirá uma interação mais dinâmica entre a aplicação e o usuário, fazendo com que ele se sinta à vontade ao utilizar o sistema, sem precisar esperar os longos intervalos de tempo entre as interações.

Dentre as principais bibliotecas utilizadas, pode-se destacar:

[26] Metodologia utilizada para a concepção do layout, ao contrário do que muitos pensam, não prima pela eliminação total do uso da tag <table> do HTML, mas sim, utilizá-las apenas para a exibição de dados tabulares e não com o objetivo de posicionar os elementos para a concepção do layout, mais informações em http://www.tableless.com.br

- *Prototype*[27]– Utilizada para a interação com a DOM, e por oferecer abstrações no uso de AJAX.

- Script.aculo.us – desenvolvida utilizando o framework Prototype, oferece ricos recursos visuais, *drag and drop*, efeitos de animação em elementos da interface, scripts de auto-completar.

Conforme Prototype (2006), a *Protoype*, é um framework JavaScript que busca facilitar o desenvolvimento de páginas Web dinâmicas. Fornecendo possibilidades de um desenvolvimento orientado a classes, *Prototype* é rápida e vem se tornando cada vez mais um padrão e a escolha por grande parte dos desenvolvedores de aplicações Web.

4.2.4 ADODB

A ADODB é um conjunto de classes que facilitam o desenvolvimento de aplicações que precisam interagir com banco de dados, tornando o desenvolvimento mais fácil e trazendo uma maior transparência no uso de diferentes SGBD's.

Conforme ADODB (2006), as funções de acesso a banco de dados do PHP não são padronizadas, isso traz a necessidade do uso de uma biblioteca de classes para esconder essas diferenças. Na verdade o que é feito são abstrações que permitem alternar entre os bancos de dados, sem que haja a necessidade de reescrever todas as funcionalidades que acessam o banco de dados.

4.3 Arquitetura

Apresentadas as tecnologias e a ferramenta em linhas gerais, essa seção apresentará a arquitetura básica da ferramenta, apresentando a interação entre as tecnologias envolvidas, além de fornecer uma idéia geral dos recursos e funcionalidades implementadas pela mesma.

[27] Maiores informações em http://prototype.conio.net/

4.3.1 Arquitetura Geral

Essa seção apresentará o fluxo básico do funcionamento da ferramenta. Um passo a passo de como ela foi elaborada, esclarecendo como as tecnologias envolvidas cooperam entre si:

- Utilizando um navegador o usuário acessa a URL da aplicação YARG, que estará em um servidor Web com suporte a tecnologia PHP;
- Os módulos da tecnologia *server-side* são acessados, esses componentes permitirão o suporte as múltiplas fontes e base de dados, conforme visto anteriormente;
- Ao estabelecer uma conexão com uma fonte de dados específica, todas as entidades que oferecem informações são apresentadas para o usuário em forma de tabelas com colunas, essas entidades podem ser selecionadas e arrastadas para uma "área de trabalho", que conterá todas as entidades que oferecerão informações para a concepção do relatório;
- Ao soltar uma tabela na área de trabalho, a mesma é representada graficamente, e novamente, as colunas podem ser arrastadas para a área do relatório;
- Após selecionar as informações e a maneira como elas ficarão dispostas o usuário poderá pré-visualizar o relatório elaborado, permitindo uma percepção exata de como o mesmo será apresentado;
- Após a pré-visualização, o relatório poderá ser impresso ou exportado nos formatos XML, CSV e PDF.

A figura 4-1, representa graficamente, todo o fluxo descrito acima:

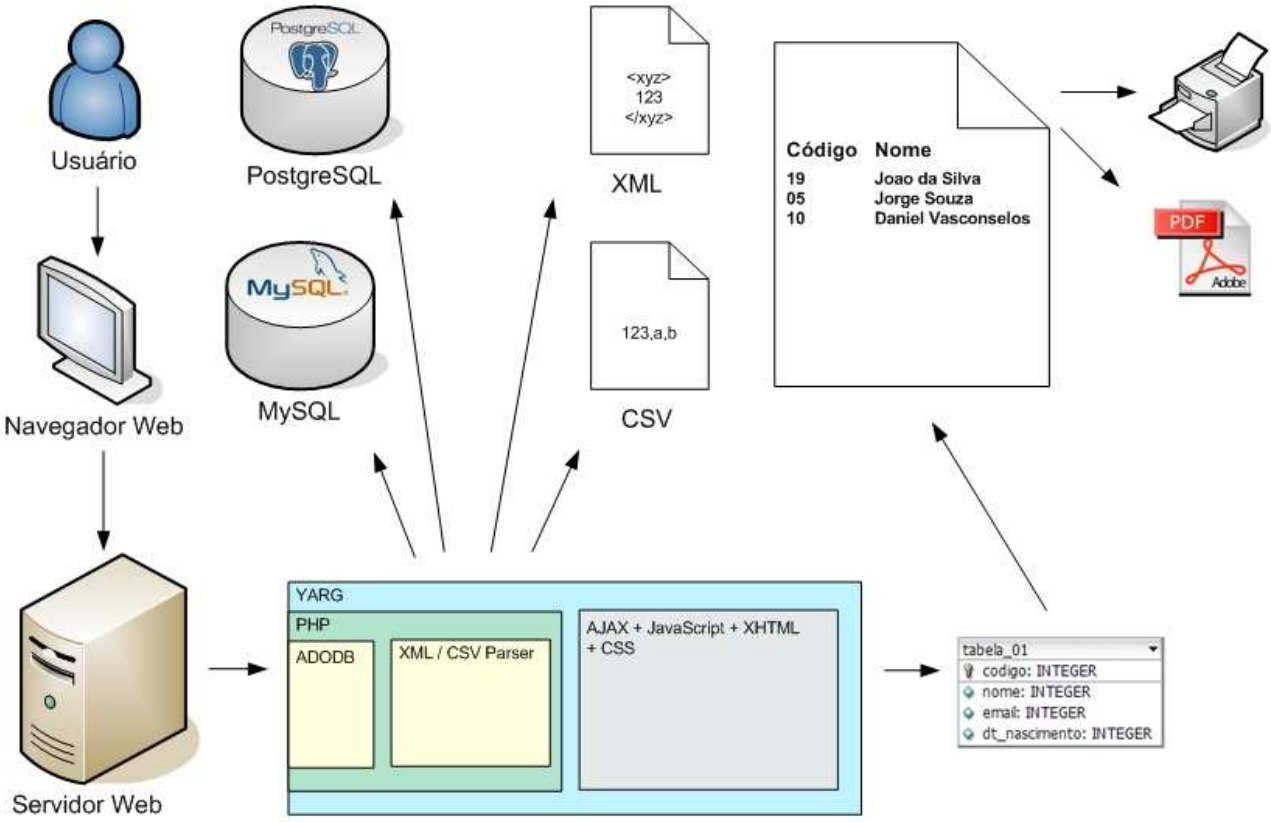


Figura 4-1 Fluxo da Arquitetura Geral

Considerações importantes:

- Não há limitação quanto ao servidor Web, pode-se usar tanto o Microsoft IIS Server, ou o Apache. O ambiente utilizado para a criação, testes e elaboração do presente trabalho foi o Apache[28];

- O PHP deve ser pré-compilado com suporte aos bancos de dados que serão utilizados de acordo com a necessidade do usuário;

Informações mais detalhadas são apresentadas na seqüência desse capítulo, para exemplificar como as tecnologias envolvidas nesse fluxo cooperam entre si.

[28] Apache, conhecido também como httpd, servidor Web de código aberto da Apache Foundation. Maiores informações podem ser encontradas em http://httpd.apache.org/

4.3.2 GUI - Interface com o usuário

As telas buscarão ao máximo seguir o padrão visual utilizado por aplicações *desktop*, de modo a trazer uma maior familiaridade do usuário com a ferramenta. Além da semelhança no padrão visual, serão utilizadas algumas funcionalidades e efeitos que até pouco tempo não eram possíveis em aplicações Web, por exemplo: recursos *drag and drop* (arrasta e solta) e atualização dinâmica da interface sem necessidade de atualizar toda a página. Além dos recursos avançados de efeitos visuais e dinamicidade da interface, é necessário ter uma visualização rápida que não utilize muitos recursos da máquina do usuário. Além disso, a ferramenta deverá ser auto-explicativa, permitindo que usuários não experientes a usem e consigam gerar seus relatórios básicos.

Dentre os recursos utilizados, além do já citado *drag and* drop, destacam-se: a possibilidade de redimensionar as sub-seções (que serão apresentadas em seguida)da tela principal da ferramenta, o uso de barra de ferramentas que podem ser movidas e posicionadas sem atrapalhar a definição do relatório.

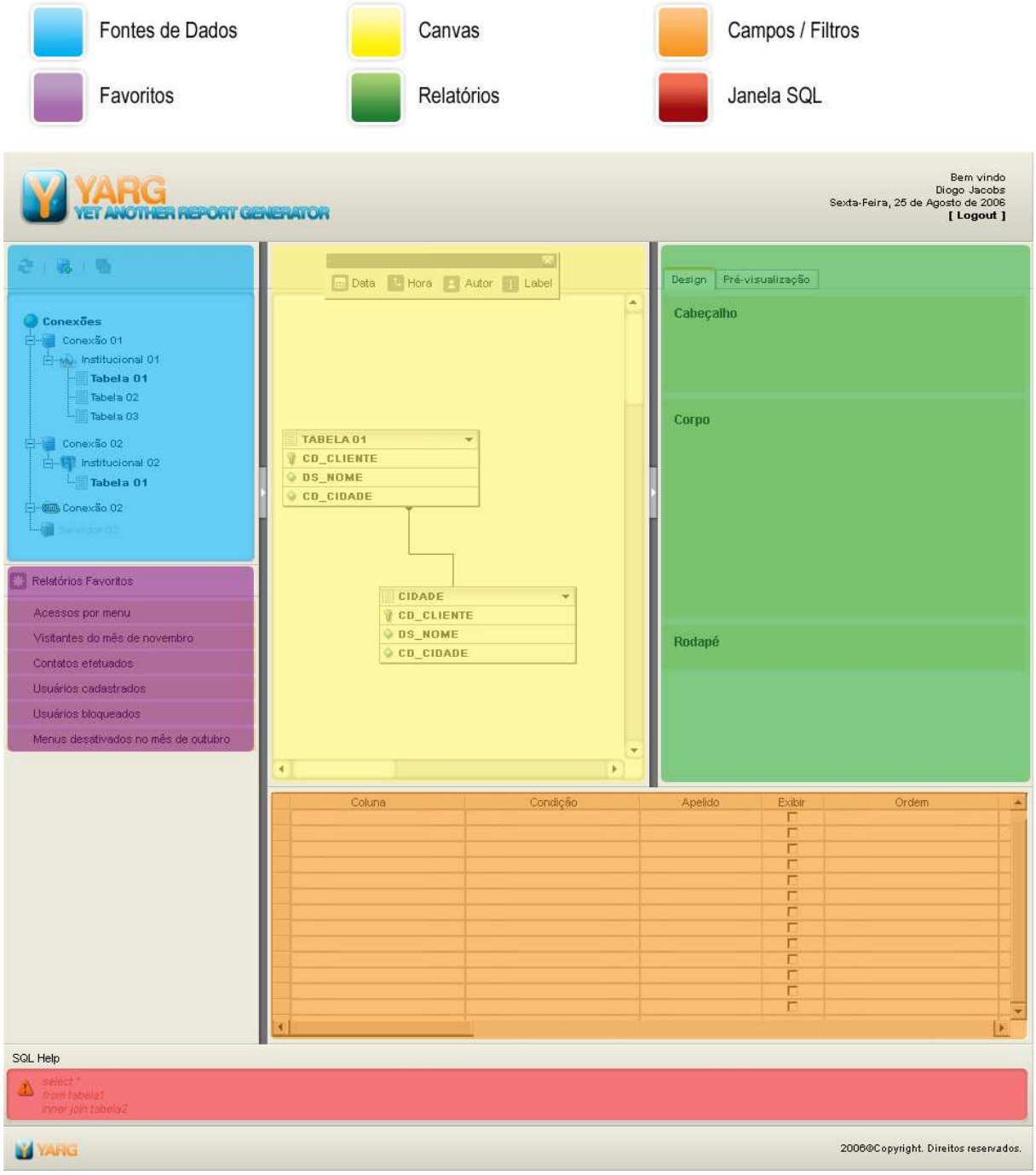


Figura 4-2 Janela principal

A figura 4-2 apresenta as subseções da janela principal do YARG, cada área tem uma "responsabilidade" específica e de acordo com o momento durante o ciclo de criação do relatório é ou não necessária a exibição de algumas delas. Por exemplo: Após definidas as tabelas que servirão como fonte das informações do relatório, pode-se optar por redimensionar ou até mesmo esconder a subseção fonte de dados, permitindo assim, uma área maior para a subseção relatório.

Para uma correta compreensão das responsabilidades de cada subseção, será apresentada em seguida, uma descrição de cada uma delas, facilitando o entendimento e estabelecendo o limite de cada subseção, permitindo assim, conhecer melhor a estrutura da ferramenta.

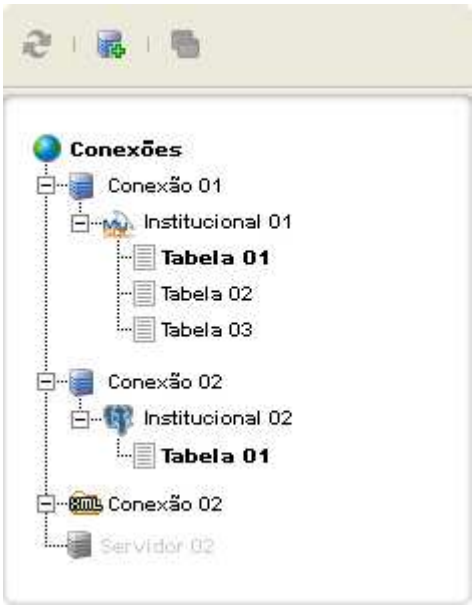


Figura 4-3 - Subseção Fontes de Dados

A subseção fontes de dados, tem como objetivo apresentar todas as conexões criadas pelo usuário, sejam elas com bancos de dados, arquivos XML ou CSV. Como mostrado na figura 4-3, foi adotada uma diferenciação entre os tipos de conexões, trata-se de uma representação gráfica, que permita ao usuário identificar qual é o tipo da fonte de dados em questão. A tabela 4-2 apresenta detalhadamente o significado de cada imagem:

Tabela 4-2 Descritivo dos tipos de conexões

Imagem	Descrição
	Refere-se a uma conexão com um servidor de banco de dados PostgreSQL
	Refere-se a uma conexão com um servidor de banco de dados MySQL
	Refere-se a uma fonte de dados, adquirida de um arquivo CSV

Refere-se a uma fonte de dados, adquirida de um arquivo XML

Todas as conexões, bem como os dados para ela, são armazenados no banco de dados da aplicação, e sempre que o usuário autentica na aplicação, as mesmas são carregadas e disponibilizadas na subseção fontes de dados.

A qualquer momento pode-se criar uma nova conexão, basta clicar no ícone de nova conexão, o assistente de criação de conexões será chamado, permitindo que o usuário forneça os dados como:

- Conexão com banco de dados: *Host* (pode ser o endereço ip ou o nome), porta, *default database*, usuário e senha;
- Arquivo XML ou CSV: O arquivo, e o nome que será dado para a fonte de dados (será apresentado com maiores detalhes em seguida).

Após informar os dados da conexão, a aplicação automaticamente a vincula com o usuário logado, possibilitando assim, que na próxima vez que ele usar a aplicação a conexão esteja disponível.

Toda a conexão apresentada na lista, está apta a ser utilizada, ao clicar em uma delas, o YARG irá se conectar, se obtiver sucesso, todas as entidades que podem fornecer dados para a aplicação são mostradas. Casos de não sucesso na conexão, não são limitações da ferramenta, mas sim indisponibilidade da fonte, no momento da tentativa.

As entidades apresentadas, podem ser utilizadas para a concepção do relatório, basta clicar sobre ela e arrastá-la para a subseção Canvas, que será considerada a área de trabalho, onde juntamente com a subseção Campos / Filtros, darão todas as opções para definir os critérios e as informações que serão exibidas no relatório.

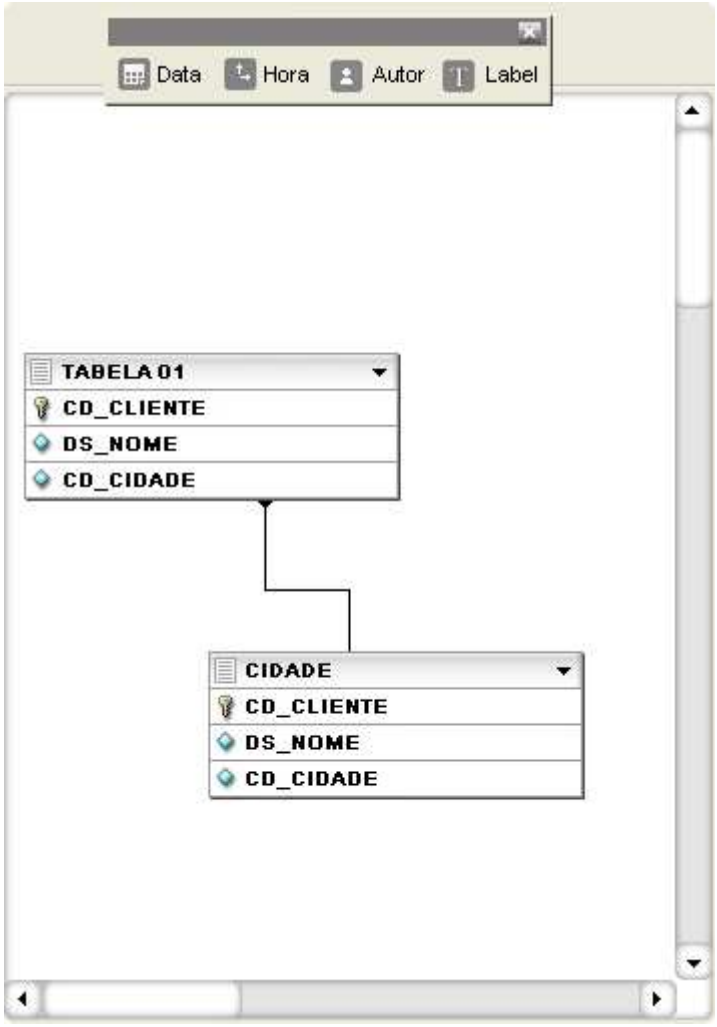


Figura 4-4 - Subseção Canvas

Como apresentado anteriormente, a subseção Canvas, é a área onde estarão todas as entidades que fornecerão informações para o relatório. Ao clicar sobre uma entidade e arrastá-la até essa área, a entidade é desenhada, também são apresentadas as colunas e a chave primária da mesma.

Toda a entidade representada na seção Canvas, permite que o usuário clique sobre um campo, e o arraste para a subseção Relatório, campos esses que automaticamente vão para a lista de campos da subseção Campos / Filtros, e também servem como base para criar as instruções SQL que serão utilizadas para buscar as informações e formar o relatório, no momento da pré-visualização, as tabelas, campos e filtros são lidos, interpretados e as instruções são geradas.

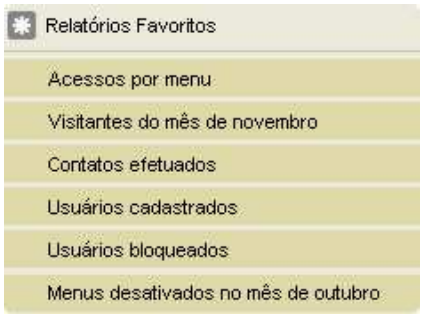


Figura 4-5 Subseção Favoritos

Essa seção conterá uma lista com links para todos os relatórios cadastrados pelo usuário, basta clicar sobre um item da lista para que todas as definições do relatório sejam carregadas. Nesse caso, as entidades na subseção canvas, filtros estabelecidos, layout do relatório serão carregadas, bastando para o usuário, apenas pré-visualizar o relatório, tendo uma noção exata de quando parou o trabalho.

Figura 4-6 - Subseção Relatórios

Conforme visto anteriormente, os campos das entidades, podem ser adicionados a essa seção, essa é a área que permitirá ao usuário definir a estrutura do relatório, possibilitando posicionar as colunas, os *labels*, as informações estáticas.

É subdivida em três áreas:

- Cabeçalho – Permite que seja definido o cabeçalho do relatório, normalmente utilizado para a exibição de informações gerais, por exemplo: data da geração, autor, quantidade de registros, entre outras;

- Corpo – Forma o relatório propriamente dito, interage diretamente com as entidades que fornecerão as informações, para cada registro / linha da entidade, uma linha é representada, seguindo o *layout* estabelecido pelo usuário.

- Rodapé – Assim como o cabeçalho, é uma área estática. Será mostrada apenas para finalizar o relatório.

Após definir a estrutura do relatório, basta clicar sobre a aba Pré-visualização, conforme mostra a figura 4-7.

Figura 4-7 Aba Pré-Visualização

Ao clicar na aba pré-visualização, o relatório é apresentado para o usuário, sendo possível, exportar o relatório em diferentes formatos ou voltar para o módulo de definição e estruturação, permitindo assim realizar ajustes na estrutura e visualizá-lo novamente.

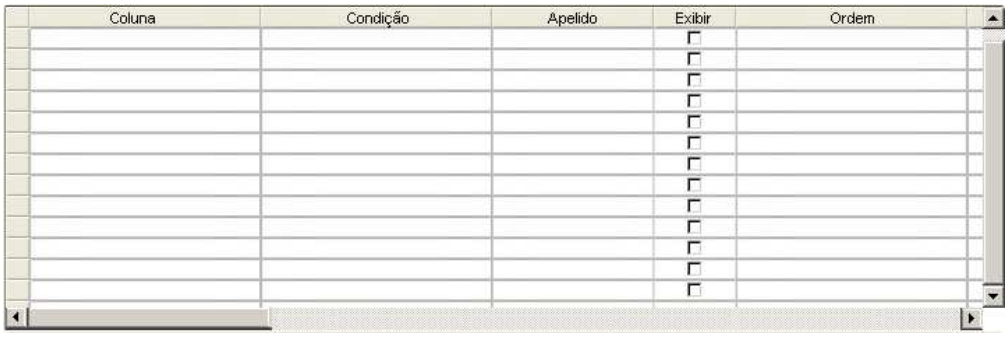


Figura 4-8 - Subseção Campos / Filtros

É possível montar um relatório simples, através da escolha das tabelas e dos campos que serão apresentados. Porém, algumas vezes, é necessário estabelecer critérios e filtros para delimitar as informações que serão apresentadas, é através dessa subseção que o usuário pode interagir com esses itens do relatório.

Conforme apresentado anteriormente, todos os campos que forem adicionados à estrutura do relatório, serão automaticamente adicionados à lista localizada na subseção Campos / Filtros, permitindo assim, que o usuário estabeleça filtros e condições, limitando as informações mostradas no relatório.

Para acrescentar um filtro, basta selecionar a condição a ser satisfeita, na coluna Condição da linha correspondente ao campo que será filtrado. A figura 4-9, representa graficamente a adição de um filtro:

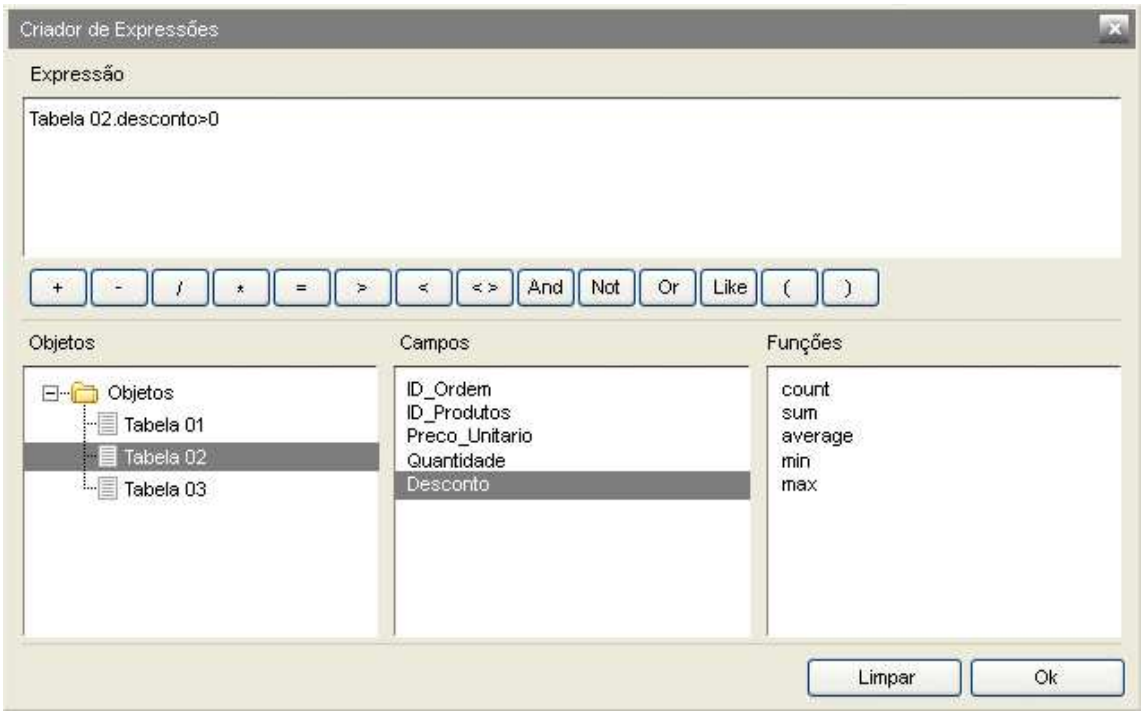


Figura 4-9 – Assistente de expressões

O YARG, tem condições de pré-validar as informações digitadas em um filtro, por exemplo, só será possível digitar caracteres numéricos em um campo do tipo numérico, evitando um erro, caso letras sejam digitadas para estabelecer um filtro em um campo numérico.

Figura 4-10 - Subseção Janela SQL

A seção Janela SQL, mostra as instruções SQL que serão executadas para realizar a busca das informações nas fontes de dados. É apenas para a visualização, ou seja, as

informações são somente leitura, não é possível interagir com as instruções esperando que as alterações reflitam no relatório e na subseção Canvas.

Não houve tempo hábil para a implementação dessa interação, uma vez que, há um certo nível de complexidade, de modo a evitar que os usuários digitam instruções incorretas, ocasionando erros na geração do relatório, ou até inviabilizando a sua execução.

Apresentada a estrutura básica da tela principal da ferramenta, faz-se necessário a apresentação de alguns recursos disponibilizados para obter-se um alto grau de semelhança com aplicações *desktop*, dentre esses recursos, pode-se citar: o uso de barra de ferramentas que podem ser escondidas e posicionadas no melhor local.

Figura 4-11 Barra de ferramentas

A Figura 4-11 apresenta a barra de ferramentas com os principais controles que serão utilizados para adicionar informações estáticas no relatório, abaixo é apresentado o significado de cada item:

- Data –Exibe a data da geração do relatório;
- Hora – Exibe a hora de geração do relatório;
- Autor – Exibe o nome do usuário que executou o relatório;
- Label – Permite adicionar textos fixos, que podem ter seu conteúdo definido pelo usuário, serve para identificar, o significado de determinada informação exibida no relatório.

O objetivo dessa seção foi apresentar em linhas gerais os recursos da interface com o usuário, exemplificando o funcionamento de cada componente da tela.

4.3.3 Independência de banco e fontes de dados

Tão ou mais importante que a interface com o usuário, a independência de banco de dados é fundamental para atender as diferentes necessidades encontradas nas organizações.

Para viabilizar a conectividade com diferentes SGBD's, foi adotado o uso de uma biblioteca / *framework*, que auxilie nos processos de comunicação com o SGBD, abstraindo as diferenças e particularidades de cada um deles, permitindo uma interação transparente entre a ferramenta e o banco.

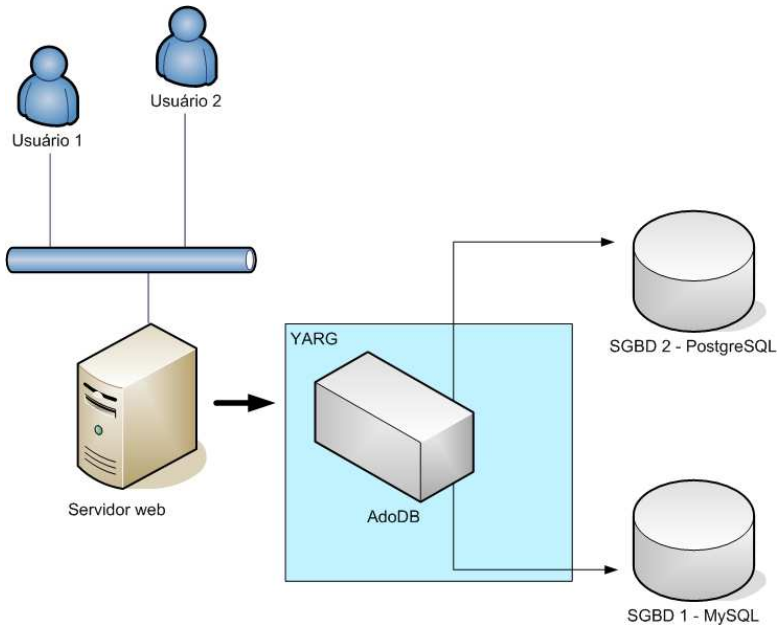


Figura 4-12 - Abstração de banco de dados

O uso da biblioteca ADODB permite que o YARG comunique-se com o banco de dados transparentemente, sem se preocupar com as diferenças entre o funcionamento dos diferentes SGBD´s suportados pela aplicação.

Além de fornecer meios de conectividade com diferentes SGBD's adotou-se a possibilidade do uso de importação de dados de arquivos estruturados, por exemplo: XML, CSV. Através de um processo de tratamento e importação e transformação desses dados, obtém-se uma estrutura auxiliar que será interpretada pela ferramenta para a geração do relatório, esse processo é explicado com maiores detalhes na seção seguinte.

Optou-se por fornecer conectividade a outras fontes de dados, para viabilizar a integração com outros sistemas, que por algum motivo não são diretamente acessíveis pela ferramenta. Por exemplo, uma forma de armazenamento de dados proprietária que não siga os padrões dos SGBD's, ou um SGBD que não esteja acessível via Web, impossibilitando a conexão com o mesmo. Dessa forma, mesmo não tendo acesso direto ao SGBD, seria possível utilizar as informações contidas no sistema em questão, através da exportação em um dos formatos suportados pela ferramenta (XML ou CSV).

4.3.4 Uso de XML e CSV como fonte de dados

O processo de importação de arquivos estruturados foi desenvolvido especialmente para viabilizar a transformação de dados contidos em arquivos em uma estrutura que o YARG possa interagir através do uso da linguagem SQL.

Ao cadastrar uma nova conexão, o sistema disponibiliza diferentes fontes de dados, dentre elas, SGBD's e arquivos. Ao selecionar uma fonte de dados que indica a necessidade de importação de um arquivo, o sistema ao invés de pedir os dados de conexão com o SGBD, irá solicitar três informações, conforme apresentado abaixo:

Figura 4-13 - Fonte de dados XML

Ao confirmar o cadastro da conexão, clicando no botão OK, o YARG irá ler todo o conteúdo do arquivo, essa informação será gravada no banco de dados em uma coluna do tipo

text, sem sofrer nenhuma tratamento especial, e utilizada para gerar um *hash*[29], que servirá para gerenciar o controle de alterações do arquivo, caso você atualize a conexão com um arquivo idêntico ao já cadastrado na base de dados, o sistema irá detectar, devido a igualdade dos *hashs* e não realizará a importação novamente. O mesmo ocorrendo para o contrário, caso você atualize a conexão fornecendo um arquivo diferente do já armazenado, o sistema irá detectar e controlar as alterações.

A conexão criada é apresentada como uma conexão normal a um SGBD, ao fazer uso da conexão, o sistema irá mostrar uma tabela criada automaticamente, com o nome do resultado do *hash* obtido anteriormente, para ficar mais legível e auto-explicativo para o usuário, o nome apresentado para a tabela, é a informação fornecida no campo "Nome para exibição", no momento do cadastro da conexão. O arquivo XML é lido, processado, os registros são extraídos e inseridos na tabela recém criada. Deste ponto em diante, o trabalho com as informações é transparente, não importa mais, se os dados são provenientes de um arquivo ou de um SGBD específico, as consultas SQL são executadas normalmente como se estivesse trabalhando com uma tabela de um banco de dados.

4.4 Modelo Relacional

O modelo relacional da aplicação será apresentado e descrito nessa seção, não é um modelo muito complexo devido a ferramenta não apresentar necessidades específicas de armazenamento de dados, devido a natureza da aplicação, trabalhar com fontes de dados externas. O modelo foi dividido em subseções, para subdividir as tabelas em pequenos grupos de responsabilidades semelhantes. Na subseção Usuário / Preferências, tem-se as tabelas usuario, usuario_x_relatorio, usuario_x_conexao. Na tabela usuario as coluas são:

- cd_usario – chave primária da tabel, portanto será o identificador único do usuário, será utilizada para fazer a associação com as preferências do usuário;
- ds_login – nome amigável ou apelido para o usuário;
- ds_nome_completo – nome_completo do usuário;

[29] Representação matemática de uma *string*, obtida através de algoritmos, cujo fim exclusivo é a geração de um código único para a mesma.

- ds_email – email do usuário, em combinação com a senha de acesso, servirá como chave para o usuário efetuar login na aplicação;

- ds_senha_acesso – senha do usuário criptografada utilizando o algoritmo SHA1;

- dt_ultimo_acesso – data do último acesso, é atualizada a cada vez que o usuário faz o login na aplicação.

A tabela usuario_x_relatorio, é o meio de ligação entre o usuário e os relatórios da seção favoritos, descrita anteriormente, tem-se as seguintes colunas:

- cd_usuario – serve como ligação com a tabela de usuários, sempre apontará para o identificador de um usuário na tabela usuario;

- cd_relatorio – serve como ligação com a tabela de relatórios, sempre apontará para um registro de um relatório na tabela relatório.

A tabela usuario_x_conexao, permite a ligação entre um usuário e as conexões criadas por ele, tem as seguintes colunas:

- cd_usuario – serve como ligação com a tabela de usuários, sempre apontará para o identificador de um usuário na tabela usuario;

- cd_conexao – serve como ligação com a tabela de conexões, sempre apontará para um registro de uma conexão na tabela conexão.

A subseção Relatório, apresenta todas as tabelas envolvidas no armazenamento das definições do relatório no banco de dados, é formada pelas tabelas: relatorio, tipo_secao, relatorio_secao. A tabela relatório é composta pelas colunas:

- cd_relatorio – Chave primária da tabela, identificador único de um relatório;

- cd_autor – Referenciará um registro da tabela usuario, identificando o autor do relatório;

- nm_relatorio – Nome amigável dado para o relatório, será essa informação que irá identificar o relatório na lista de favoritos;

- dt_cadastro – Data de criação do relatório;
- dt_modificacao – Data da última alteração realizada no relatório;
- dt_ultima_execução – Data da última execução do relatório.

A tabela tipo_secao, armazenará os tipos de seções de um relatório, inicialmente serão apenas as três seções básicas: cabeçalho, corpo e rodapé, é formada pelas seguintes colunas:

- cd_tipo_secao – Chave primária da tabela, identificador único da seção, será utilizada como chave na tabela relatorio_seção;
- ds_tipo_secao – Nome dado para a seção do relatório.

A tabela relatorio_secao, será utilizada para armazenar o conteúdo de cada seção do relatório, tem as seguintes colunas:

- cd_relatorio_secao – Chave primária da tabela;
- cd_relatorio – Apontará para o relatório, será utilizada para restaurar as seções correspondentes a um determinado relatório;
- cd_tipo_secao – Identificará a qual seção o conteúdo se refere;
- tx_conteudo_secao – Conteúdo da seção, será armazenado utilizando um formato pré definido, conforme explicado na seção 4.6.3.

A subseção Conexão apresenta as tabelas envolvidas no armazenamento de conexões com bancos de dados e arquivos externos, é formada pelas tabelas: conexao, conexao_banco, conexao_arquivo, sgbd. A tabela conexão é composta pelas colunas:

- cd_conexao – Chave primária da tabela, será utilizada como chave para vincular as conexões com um usuário;
- cd_sgbd – Identifica o tipo de conexão (SGBD ou arquivo);
- ds_conexao – Nome dado para a conexão no momento da criação;

- dt_cadastro – Data de criação da conexão.

A tabela conexao_banco, irá armazenar as configurações da conexão com o SGBD, é formada pelas seguintes colunas:

- cd_conexao – Apontará para um registro da tabela conexao;
- ds_host – O nome do host ou o endereço ip do servidor ao qual deseja-se conectar;
- nr_porta – Porta que será utilizada para realizar a conexão com o SGBD;
- ds_usuario – Nome do usuário que conectará no banco de dados;
- ds_senha – Senha do usuário informado na coluna ds_usuario;
- ds_banco – Nome do database inicial (*default database),* banco de dados que será utilizado ao realizar a conexão.

A tabela conexao_arquivo, irá armazenar as informações de uma conexão com um arquivo XML ou CSV, é formada pelas seguintes colunas:

- cd_conexao – Apontará para um registro da tabela conexao;
- ds_nome_tabela – Armazenará o nome atribuído a tabela criada com os dados importados do arquivo;
- tx_conteudo_arquivo – Armazenará o conteúdo do arquivo importado.

A tabela sgbd, armazenará os tipos de fontes de dados, poderá armazenar parâmetros referentes a um fornecedor específico do SGBD: MySQL, PostgreSQL, ou referencias para os módulos de importação de arquivos, é formada pelas colunas:

- cd_sgbd – Chave primária, identificador único dos tipos de fontes de dados;
- ds_sgbd – Nome do tipo de fonte de dados;

- nr_porta_padrao – Porta padrão do SGBD em questão para ser usada como sugestão no momento da criação da nova conexão;

- ds_usuario_padrao – Nome do usuário padrão que se conectará ao SGBD, também será utilizado como sugestão, na criação de uma nova conexão;

- id_modulo_importação – Identificador do módulo que será utilizado para criar a conexão, identifica se é a importação de um arquivo ou a conexão a um SGBD, no caso de ser uma conexão, identifica o identificador utilizado na ADODB para criar a conexão.

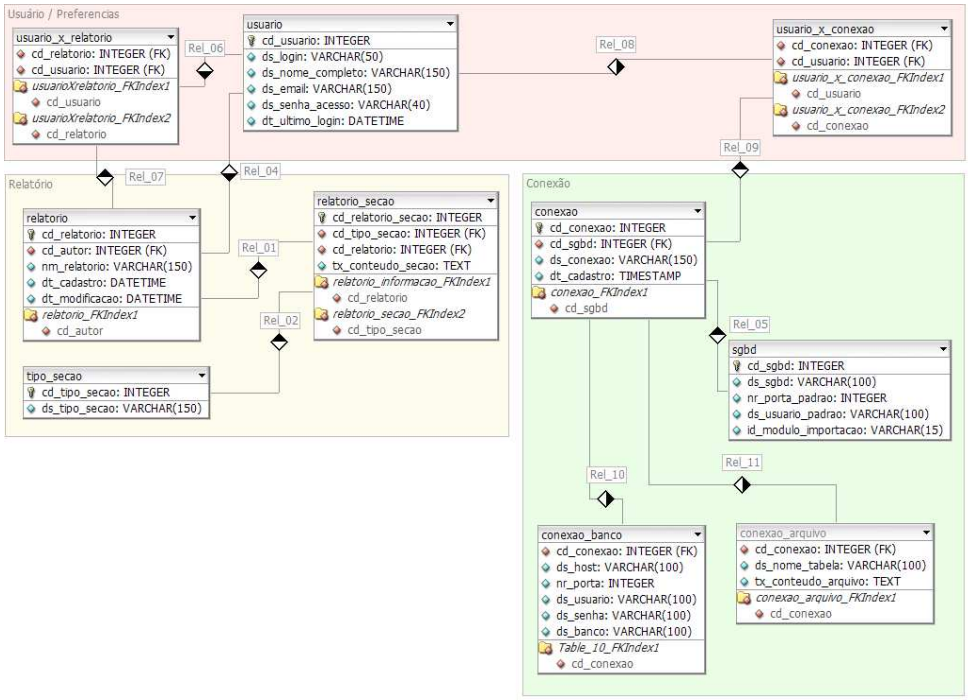


Figura 4-14 - Modelo Relacional

O objetivo dessa seção foi apresentar o modelo relacional da aplicação, descrevendo a estrutura das tabelas e os dados que serão armazenados em cada uma.

4.5 Funcionalidades

Por se tratar de uma aplicação que na maioria das vezes, não será executada na máquina do usuário, certos conceitos foram adotados para viabilizar um ambiente virtual, único, permitindo assim simular a área de trabalho e até mesmo o computador pessoal de cada usuário. Ao autenticar-se, fornecendo um e-mail e uma senha, o sistema irá buscar um usuário

com as informações fornecidas, caso encontre, buscará automaticamente as preferências do usuário (conexões e relatórios anteriormente criados). A figura 4-14 demonstra o funcionamento desse procedimento:

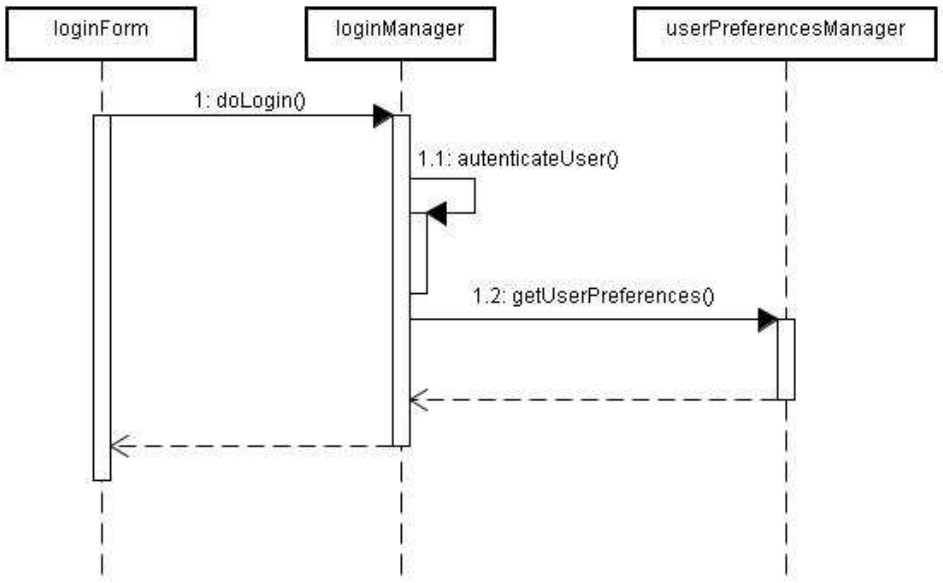


Figura 4-15 - Obtenção das preferências do usuário

Após buscar as preferências do usuário, a tela principal é mostrada, com as subseções totalmente vazias e sem informações, permitindo que um relatório seja carregado, ou que um novo seja elaborado.

Para um correto entendimento do funcionamento da ferramenta, foram elaborados dois exemplos, um trabalhando com um SGBG PostgreSQL e outro trabalhando com um arquivo XML.

4.5.1 Trabalhando com um SGBD PostgreSQL

O departamento de marketing de uma empresa, precisa apresentar um relatório para o diretor da companhia, mostrando a quantidade de acessos de cada menu do site institucional, porém o único dado que se tem, é que esses acessos são armazenados em uma tabela do banco de dados do site e que o SGBD utilizado, é o PostgreSQL.

Primeiramente, deve-se criar uma conexão com o servidor, através do assistente de conexões do YARG. Para isso, basta clicar no ícone Nova Conexão, apresentado na figura 4-15.

Figura 4-16 Botão nova conexão

A primeira informação que o assistente de conexões solicita é o tipo da fonte de dados, nesse exemplo, será selecionada a opção PostgreSQL, após selecionado, deve-se clicar o botão próximo, para dar seqüência ao procedimento de criação de uma nova conexão, conforme mostrado na figura 4-16.

Figura 4-17 Nova conexão - Passo 1

No próximo passo, um conjunto distinto de informações é solicitado, dependendo da resposta do usuário, nesse caso, esse conjunto é formado por: nome da conexão, nome ou endereço IP do servidor, porta, usuário, senha, e o *database* inicial, conforme figura 4-17.

Figura 4-18 Nova conexão - Passo 2

Após fornecer as informações solicitadas, é possível concluir o assistente, ou efetuar um teste de conexão com o servidor correspondente aos dados fornecidos, caso haja sucesso, uma mensagem de sucesso é apresentada e ao concluir o assistente, a nova conexão será mostrada na lista de conexões do usuário.

Ao clicar sobre a conexão, o sistema conectará no servidor correspondente e irá apresentar as tabelas para o usuário. Apresentadas as tabelas, dentre elas, a tabela que será o alvo desse relatório, a que controla os acessos de cada menu, deve-se clicar sobre ela e arrastá-la para a subseção Canvas. Nesse momento têm-se as informações para a elaboração do relatório. Em seguida deve-se clicar na coluna codigo_menu, e arrastá-la até a área Corpo da subseção Relatório.

Para criar a informação de quantidade de acessos por menu, deve-se utilizar o criador de expressões, para isso, basta clicar no botão Criar Expressão da subseção Campos / Filtros, que apresentará o assistente de expressões conforme figura 4-9, apresentada anteriormente.

No caso, o objetivo é contar a quantidade de registros da tabela acesso_menu, para cada menu do sistema, sendo assim, seleciona-se a coluna codigo_menu no criador de expressões e diz-se que a operação a ser utilizada é a função *count*, para confirmar a seleção, basta clicar no botão OK, e a expressão recém criada aparecerá na lista de campos.

Como o objetivo é ter uma quebra por menu é necessário fornecer a coluna pela qual será realizado o agrupamento, que no exemplo, é a coluna codigo_menu. Ao clicar na aba

Pré-visualização, o relatório é apresentado na tela, o que se obteve até o momento é algo semelhante a:

Figura 4-19 - Relatório de acessos - Parte 1

O relatório apresentado, mostra o que se propõe, porém é pouco legível, seria melhor se ao invés do código do menu, fosse exibido o nome. Como a tabela de acessos só tem uma referência para a chave da tabela de menus, faz-se necessário o uso da tabela de menus para permitir que o nome do menu seja apresentado no relatório.

Para tanto, deve-se clicar sobre a tabela menu e arrastá-la para a subseção Canvas, visando evitar uma operação de produto cartesiano entre as duas tabelas, é necessário fornecer um critério de união entre elas, no caso a coluna codigo_menu, essa operação é feita no YARG, clicando sobre a coluna codigo_menu da tabela acesso_menu, arrastando-a até a coluna codigo_menu da tabela menu, isso fará com que ao ler a estrutura do relatório, seja gerada uma operação de *join* entre as tabelas.

A união entre as tabelas já foi feita, agora basta mover a coluna descricao_menu da tabela menu, para a área Corpo da subseção Relatório, ao clicar na aba Pré-visualização, tem-se algo semelhante a:

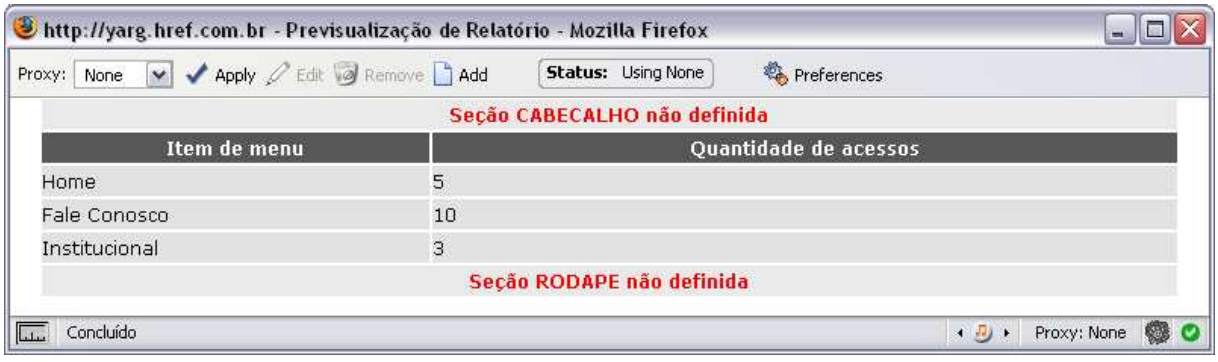


Figura 4-20 - Relatório de acessos - Parte 2

Esse foi um exemplo simples de extração de informações a partir de tabelas de um SGBD PostgreSQL. Apesar de simples, alguns recursos de agrupamento e união de tabelas foram apresentados, foi possível montar o relatório de acessos por menu, sem escrever manualmente as instruções SQL.

4.5.2 Trabalhando com um arquivo XML

Uma empresa, administradora de cartões de crédito, precisa gerar um relatório com o resumo de transações financeiras ocorridas no dia primeiro de novembro de 2006. Porém as informações das transações, não estão na base da administradora de cartões, pois trata-se de um serviço terceirizado, a empresa contratada para realizar a captura das transações financeiras envia diariamente um arquivo XML, contendo um resumo de todas as operações do dia. Dado esse cenário, essa seção visa apresentar o passo a passo para elaborar o relatório baseado nas informações do arquivo recebido.

Primeiramente, deve-se criar a conexão com a fonte de dados XML, no caso, o arquivo de transações, o funcionamento do assistente de conexão, já foi descrito anteriormente, e pode ser utilizado da mesma forma, porém ao invés de selecionar o PostgreSQL, deve-se selecionar a opção XML e clicar no botão Próximo.

As informações solicitadas no segundo passo, são diferentes das solicitadas anteriormente, nesse caso é solicitado o nome que será dado à tabela e o caminho para o arquivo XML que será importado.

Ao informar os dados solicitados deve-se concluir o processo de criação de nova conexão, a conexão recém criada, será apresentada na lista de conexões do usuário, e estará apta ao uso. Ao confirmar seu cadastro a estrutura do arquivo XML é armazenada em uma tabela do sistema, sendo assim, é possível reconstruir as informações do relatório a qualquer momento.

Assim como no procedimento do banco de dados, a entidade é apresentada, e pode ser movida para a subseção Canvas, ao soltar a tabela, o sistema realizará o *parsing* de todo o conteúdo do arquivo XML, e uma tabela será criada automaticamente, com a estrutura e dados presentes no arquivo de origem.

Desse ponto em diante, o funcionamento é exatamente igual, ao do procedimento descrito anteriormente utilizando uma fonte de dados proveniente de um SGBD, basta arrastar as colunas para a área Corpo da subseção Relatório. Ao clicar na aba Pré-visualização, o relatório com as transações do arquivo XML é apresentado na tela e pode ser exportado e apresentado para as pessoas que geraram a demanda.

O procedimento se assemelha muito ao da interação com um SGBD, devido a solução adotada para o trabalho com os arquivos XML e CSV, para não se perder os recursos avançados, oferecidos pela linguagem de consulta SQL, optou-se por realizar o *parsing* do arquivo, transformando-o em uma tabela do sistema, permitindo assim que toda e qualquer funcionalidade valida para uma tabela originalmente definida no esquema do banco de dados seja também aplicável a tabela criada, baseada na estrutura do arquivo importado.

Através desses exemplos, objetivou-se apresentar as possibilidades de uso da ferramenta, bem como a flexibilidade oferecida para a obtenção de informações e a estruturação de um relatório.

4.6 O ciclo de vida da criação de um relatório

Os exemplos apresentados anteriormente trouxeram uma idéia das etapas envolvidas para a criação de relatórios utilizando a ferramenta desenvolvida. O objetivo dessa seção é apresentar como as informações apresentadas para o usuário são armazenadas, bem como a maneira pela qual todas as ações do usuário são armazenadas para após isso ser interpretadas e gerarem o relatório.

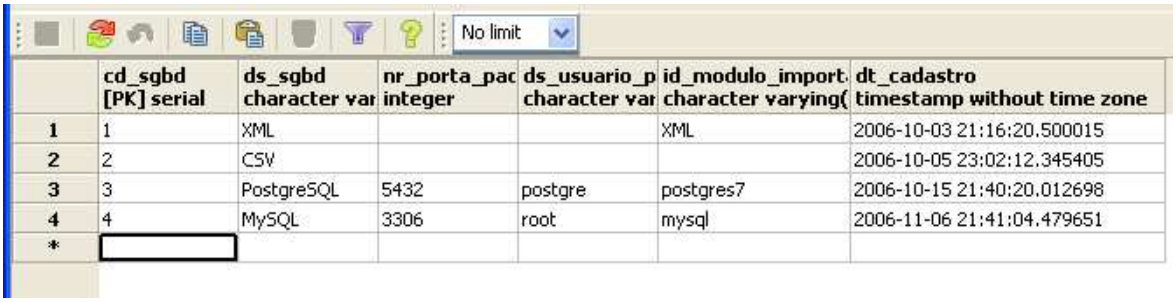
Foi adotada a seguinte metodologia para apresentação: uma breve descrição e após isso uma imagem com os registros armazenados no banco de dados do YARG, a visualização dos registros foi feita utilizando a ferramenta pgAdmin[30] 1.4.0.

[30] Aplicação que permite a administração de um banco de dados PostgreSQL, maiores informações podem ser obtidas em: http://www.postgresql.org/

4.6.1 Fontes de dados

Todas as opções apresentadas na primeira etapa de criação de conexões são armazenadas na tabela sgbd, descrita com detalhes na seção sobre o modelo relacional do YARG.

A figura 4-21 apresenta os registros da tabela sgbd, correspondente as opções apresentadas na figura 4-17:



No limit

	cd_sgbd [PK] serial	ds_sgbd character var	nr_porta_pac integer	ds_usuario_p character var	id_modulo_import character varying(	dt_cadastro timestamp without time zone
1	1	XML			XML	2006-10-03 21:16:20.500015
2	2	CSV				2006-10-05 23:02:12.345405
3	3	PostgreSQL	5432	postgre	postgres7	2006-10-15 21:40:20.012698
4	4	MySQL	3306	root	mysql	2006-11-06 21:41:04.479651
*						

Figura 4-21 – Registros da tabela SGBD

Para acrescentar uma nova opção de conexão compatível com a biblioteca ADODB, basta inserir um registro nessa tabela e identificar qual o módulo responsável pela integração com o banco de dados na coluna id_modulo_importacao. O gerenciamento dessa tabela não é possível através do YARG, por questões de segurança.

4.6.2 Conexões

Conforme apresentado até o momento, têm-se dois tipos de conexão: as conexões com SGBD's e as com arquivos de dados estruturados. Ao criar uma nova conexão um registro é inserido na tabela conexao, e dependendo do tipo na tabela conexao_banco ou conexao_arquivo.

No caso de uma conexão com um SGBD, apenas as informações de conexão com o servidor são armazenadas, a lista de tabelas é obtida no momento da conexão e fica disponível apenas enquanto a conexão estiver ativa. A figura 4-22 apresenta os registros da tabela conexao:

	cd_conexao [PK] serial	ds_conexao character var	dt_cadastro timestamp without time zone	dt_modificacao timestamp without time z	cd_sgbd integer
1	28	xml ok	2006-10-22 21:04:14.968943		1
2	29	banco 1	2006-10-22 23:01:41.319581		3
3	30	Institucional	2006-11-03 21:05:02.994483		4
4	31	Transações	2006-11-06 21:05:21.237972		3
*					

Figura 4-22 - Registros da tabela conexao

Nota-se que as informações da conexão, não estão armazenadas na mesma, mas sim na tabela conexao_banco, que tem um vínculo com a anterior através da coluna cd_conexao, conforme apresentado na figura 4-23:

	cd_conexao [PK] integer	ds_host character varying(100)	nr_porta integer	ds_usuario character varying(100)	ds_senha character varying(100)	ds_banco character varying(100)
1	29	pgsql.poloshoes.com.br	5432	poloshoes.com.br		poloshoes.com.br
2	30	mysql.tca.com.br	3306	zeket		zeket
3	31	201.7.198.236	5433	getnet		getnet.com.br
*						

Figura 4-23 - Registros da tabela conexao_banco

4.6.3 Especificação dos atributos

Além do cadastro de conexões e fontes de dados, faz-se necessário o armazenamento das informações provenientes da interação do usuário, são elas: as tabelas que estão na subseção Canvas, as junções (*joinings*) entre as tabelas, os filtros, as colunas e resultados de funções de agregação, as instruções de ordenamento e agrupamento das consultas.

Todas essas informações são armazenadas em forma de uma árvore estruturada hierarquicamente, em formato XML e ficam na coluna tx_conteudo_secao da tabela relatorio_secao. A especificação da estrutura XML utilizada é abordada em seguida com um exemplo prático.

Considerando o exemplo criado anteriormente, o relatório que apresenta a quantidade de acessos de um determinado menu, tem-se a seguinte definição do relatório:

```
<relatorio>
  <idConexao>10</idConexao >
  <tabelas>
    <tabela id="menu">
```

```
      <position>
        <x>15</x>
        <y>10</y>
      </position>
      <joins>
        <join id="menu_x_acesso_menu">
          <joining>cd_menu</joining>
          <joined>acesso_menu.cd_menu</joined>
          <method>inner</method>
          <on>$1=$2</on>
        </join>
      </joins>
      <where>
        <group id="1">
          <boolean>and</boolean>
          <fields>
            <field id="0">
              <fieldname>menu.cd_menu</fieldname>
              <operator><></operator>
              <criteria>NULL</criteria>
            </field>
            <field id="1">
              <fieldname>menu.dt_desativacao</fieldname>
              <operator><></operator>
              <criteria>NULL</criteria>
            </field>
          </fields>
        </group>
      </where>
    </tabela>
    <tabela id="acesso_menu">
      <position>
        <x>120</x>
        <y>10</y>
      </position>
    </tabela>
  </tabelas>
  <estrturaRelatorio>
    <shownFields>
      <showfield id="0">
        <name>menu.ds_menu</name>
        <alias>Item de menu</alias>
```

```
        <order/>
        <section>corpo</section>
      </showfield>
      <showfield id="1">
        <name>acesso_menu.cd_menu</name>
        <alias>Quantidade de acessos</alias>
        <order>DESC</order>
        <agg>COUNT</agg>
        <section>corpo</section>
      </showfield>
    </shownFields>
  </estrturaRelatorio>
</relatorio>
```

Toda essa estrutura é lida e interpretada gerando a seguinte consulta SQL:

```
select      menu.ds_menu as field_00,
            count(acesso_menu.cd_menu) as field_01
from        menu
                  inner join acesso_menu
                        on menu.cd_menu = acesso_menu.cd_menu
where       (menu.cd_menu <> NULL and menu.dt_desativacao <> NULL)
group by menu.ds_menu
```

Ao receber o resultado da consulta SQL, o YARG extrai o identificador fornecido para as colunas, no caso, field_00 e field_01, realiza uma pesquisa na estrutura para buscar informações da seção shownFields como por exemplo, o *label* que será mostrado na coluna do relatório correspondente ao identificador da coluna da consulta SQL Essa seção pode ser usada no futuro para aplicar formatações condicionais e máscaras durante a exibição de uma determinada informação.

Ao salvar o relatório registros serão criados nas tabelas: relatorio, relatorio_secao, as informações descritas anteriormente ficaram em um registro na tabela relatorio_secao, correspondente a seção Corpo do relatório salvo.

A maneira adotada para o armazenamento das definições do relatório, traz uma grande flexibilidade para a continua evolução da ferramenta, possibilitando o agregamento de novas funcionalidades sem haver a necessidade de alteração do banco de dados do YARG. Novas propriedades podem ser adicionadas, e a ferramenta só terá que interpreta-las da

maneira correta, não interferindo em relatórios que por ventura não tenham a definição de determinada propriedade ou atributo.

CONCLUSÃO

Um gerenciador de relatórios disponibilizado em forma de aplicação Web consegue obter o alto grau de flexibilidade e agilidade necessário no ambiente corporativo anteriormente definido e contextualizado. A ferramenta desenvolvida busca atender as necessidades de usuários não integrantes da equipe de desenvolvimento do software.

Nos primeiros capítulos desse trabalho, o objetivo principal foi apresentar o problema que a ferramenta desenvolvida se propôs a resolver: permitir a elaboração de relatórios customizáveis em um ambiente altamente disponível, a Web. Além disso, foi apresentada a mudança de comportamento no desenvolvimento de aplicações, ilustrando esta evolução através de exemplos das bibliotecas e frameworks que surgiram nos últimos tempos, principalmente o AJAX.

Uma vez contextualizado, dissertado a respeito dos problemas encontrados na maioria das aplicações desenvolvidas nos dias atuais e apresentada a importância da geração de relatórios customizáveis, partiu-se para análise, projeto e desenvolvimento de uma ferramenta que atendesse as demandas de criação de relatórios a partir de um ambiente Web e principalmente, que permitisse a obtenção de dados de diferentes SGBD's.

Durante o desenvolvimento da mesma, percebeu-se que a indisponibilidade de algumas informações, dada as diversas limitações dos sistemas, dentre elas: indisponibilidade de conexão a partir da Web, uso de sistema de banco de dados incompatível com o padrão SQL, entre outros, inviabiliza a extração e elaboração de um relatório. Considerando esse fato, foram desenvolvidos módulos capazes de realizar a importação de arquivos XML e CSV

e convertê-los para uma estrutura auxiliar que viabilizasse o uso de SQL nos dados oriundos dos arquivos.

Algumas limitações na interação entre o PHP e alguns SGBD's foram encontradas, por exemplo: a conectividade a bancos de dados Sybase, só é possível utilizando uma biblioteca especial, chamada FreeTDS. Portanto, é necessário interagir com arquivos de configuração dessa biblioteca para viabilizar a conectividade, foi estudada a possibilidade de alteração dos arquivos de configuração, mas não houve tempo hábil para finalizar essa implementação, o que pode ser realizado em um trabalho futuro.

Outro ponto que mereceu especial atenção durante o desenvolvimento, foi a incompatibilidade das funcionalidades entre os navegadores testados, Internet Explorer e Mozilla Firefox, sendo utilizadas, as versões 6.0 e 2.0 respectivamente. As incompatibilidades eram de conhecimento do aluno e do orientador, porém os recursos avançados da ferramenta, aguçaram consideravelmente a complexidade de manter o padrão, mesmo utilizando os *guidelines* especificados pelo W3C. O trabalho ficou 100% operacional no navegador Mozilla Firefox. O navegador Internet Explorer, apresenta pequenos problemas no tratamento dos eventos do botão do mouse, o ajuste da aplicação visando oferecer compatibilidade entre todos os navegadores também pode ser realizado em um trabalho futuro.

Apesar de permitir a geração de relatórios com dados oriundos de diferentes fontes de dados, a continuidade do projeto é inevitável, havendo possibilidade para extensão em diversos aspectos, sendo do interesse do orientador e do aluno. Sendo assim, é deixado um espaço para uma série de trabalhos futuros, que ou não estavam no escopo da ferramenta, ou foram fatores complicadores durante o desenvolvimento da mesma, são eles: conectividade ao SGBD Sybase, maior interatividade com arquivos XML, disponibilidade para diferentes tipos de relatório, elaboração de gráficos.

REFERÊNCIAS BIBLIOGRÁFICAS

ADODB. ADODB Manual, 2006. Disponível em: <http://phplens.com/lens/adodb/docs-adodb.htm#intro>. Acesso em: 15 nov. 2006.

BELLO, D. D. A "alma" do hipertexto, 2002. Disponível em: <http://reposcom.portcom.intercom.org.br/bitstream/1904/19147/1/2002_NP15bello.pdf>. Acesso em 21 jun. 2006.

BERVIAN, A. E. **CRITÉRIOS PARA A DECISÃO DE PERSONALIZAÇÃO DE PROCESSOS NA IMPLANTAÇÃO DE SISTEMAS ERP**, 2004, Monografia (Bacharel em Informática) – Unisinos, São Leopoldo.

BOGUI, C.; SHITSUKA, R. **Sistemas de Infomração: Um enfoque dinâmico**.1ª ed. São Paulo: Érica, 2002.

BUSINESS OBJECT **Reporting**,2006a. Disponível em: <http://www.businessobject.com.br/reporting.asp>. Acesso em: 13 jun. 2006.

BUSINESS OBJECT **Crystal Reports para Criação de Relatórios Corporativos**, 2006a. Disponível em: <http://www.brazil.businessobjects.com/produtos/reporting/crystalreports/default.asp>. Acesso em 21 jun. 2006.

CASTRO,R.S. **Eclipse BIRT – Uma nova opção para relatórios**, 2006, Mundo JAVA Ed 16, p.10.

DALL'OGLIO, P. **Um estudo sobre o desenvolvimento e utilização de Sistemas para Geração de Relatórios**, Wcompi-Univates,2002.

DATE, C. J. **Introdução a Sistemas de BANCO DE DADOS** Rio de Janeiro: Elsevier, 2003.

ELMASRI,R.; NAVATHE S. B. **Sistemas de Banco de Dados** São Paulo: Addison Wesley, 2005.

GARRET J. J. **AJAX: A New Approach to Web Applications**, 2005. Disponivel em: <http://www.adaptivepath.com/publications/essays/archives/000385.php>. Acesso em 18 jun. 2006.

IBM. **AJAX – When to consider as a viable RIA technology**, 2006. Disponível em: <http://www-128.ibm.com/developerworks/forums/dw_thread.jsp?forum=786&thread=111946&cat=67>. Acesso em 18 jun. 2006.

INTERNET Pionners – Vannevar Bush, 2006. Disponível em: <http://www.ibiblio.org/pioneers/bush.html>. Acesso em 20 jun. 2006.

JENSEN, J. R. **AJAS performance stats, ROI, and business value**, 2006. Disponível em: <http://justaddwater.dk/2006/01/14/ajax-performance-stats-roi-and-business-value/>. Acesso em 20 jun. 2006.

LACHEV T. **Microsoft Reporting Services in Action** USA: Manning, 2004.

MCLELLAN, D. **Very Dynamic Web Interfaces**. Disponível em: <http://www.xml.com/pub/a/2005/02/09/xml-http-request.html>. Acesso em: 17 jun. 2006.

MELONI, J. C. **PHP 5 Fast & Easy Web Development** Boston: Course PTR, a division of Course Technology, 2004.

MEYER, M.; BABER, R.; PFAFFENBERGER, B. **Nosso futuro e o computador**. 3ª ed. Porto Alegre: Bookman, 2000.

MORESI, E.A.D. **Delineando o valor do sistema de informação de uma organização**, 2000. Disponível em <http://www.ibict.br/cionline/include/getdoc.php?id=592&article=284&mode=pdf>.Acesso em 12 mai. 2006.

MOZILLA. **AJAX:Getting Started**, 2006. Disponível em: <http://developer.mozilla.org/en/docs/AJAX:Getting_Started> . Acesso em 18 jun. 2006.

SILBERSCHARTZ, A.;KORTH H. F.; SUDARSHAN S. **Sistemas de Banco de Dados** São Paulo: MAKRON Books, 1999.

STEIL, R.; CAMARGO, F. M. **Ajax – Desenvolvendo uma Web mais interativa**. Revista MUNDO JAVA. Edição 14, 2005.

SUN. **Lesson: Applets (The Java™ Tutorials > Deployment).** Disponível em: <http://java.sun.com/docs/books/tutorial/deployment/applet/index.html>. Acesso em 15 nov. 2006.

TELERIK **Using AJAX in Web Applications – A Practical Guide for ASP.NET Developers**, 2005. Disponível em: <http://www.telerik.com/documents/Telerik_and_AJAX.pdf>. Acesso em: 20 jun. 2006.

UNDERWOOD, L. **Why Switch to XHTML?**, 2004. Disponível em: < http://www.webreference.com/authoring/xhtml/ > .Acesso em 15 nov. 2006.

W3C. **About the World Wide Web Consortium (W3C)**, 2006a. Disponível em: <http://www.w3.org/consortium/overview.html>. Acesso em 18 jun. 2006.

W3C. **About W3C > Goals**, 2006b. Disponível em: <http://www.w3.org/Consortium/mission>. Acesso em 18 jun. 2006.

W3C. **The XMLHttpRequest**, 2006c. Disponível em: <http://www.w3.org/TR/XMLHttpRequest/>. Acesso em 18 jun. 2006.

WEBAPPLICATIONSOLUTIONS. **Web Application Solutions: A Designers Guide,** 2006. Disponível em: <http://www.lukew.com/resources/WebApplicationSolutions.pdf>. Acesso em 15 nov. 2006.

WHITE, A. **Measuring the Benefits of Ajax**, 2006. Diposnível em: <http://www.developer.com/xml/article.php/3554271>. Acesso em 20 jun. 2006.

WIKIPEDIA. **Internet**, 2006a. Disponível em: <http://en.wikipedia.org/wiki/Internet>. Acesso em: 18 jun. 2006.

WIKIPEDIA. **Memex**, 2006b. Disponível em: <http://en.wikipedia.org/wiki/Memex>. Acesso em: 18 jun. 2006.

WIKIPEDIA. **World Wide Web**, 2006c. Disponível em: <http://pt.wikipedia.org/wiki/World_wide_web>. Acesso em 20 jun. 2006

WIKIPEDIA. **Tim Berners-Lee**, 2006d. Disponível em: <http://en.wikipedia.org/wiki/Tim_Berners-Lee>. Acesso em 20 jun. 2006.

WIKIPEDIA. **AJAX (programming)**, 2006e. Disponível em: <http://en.wikipedia.org/wiki/AJAX>. Acesso em: 18 jun. 2006.

XULPLANET. **1.1 Introduction**, 2006. Disponível em: <http://www.xulplanet.com/tutorials/xultu/intro.html>. Acesso em: 15 nov. 2006.

ZAKON, Robert H. **Hobbe's Internet Timeline v8.1**, 2005. Disponível em: <http://www.zakon.org/robert/internet/timeline/>. Acesso em 20 jun. 2006.