CENTRO UNIVERSITÁRIO FEEVALE

DIOGO RAFAEL JACOBS

UM GERENCIADOR DE RELATÓRIOS UTILIZANDO AJAX

Novo Hamburgo, junho de 2006.

DIOGO RAFAEL JACOBS

UM GERENCIADOR DE RELATÓRIOS UTILIZANDO AJAX

Centro Universitário Feevale
Instituto de Ciência Exatas e Tecnológicas
Curso de Ciência da Computação
Trabalho de conclusão de curso

Professor Orientador: Prof. Juliano Varella de Carvalho

Novo Hamburgo, junho de 2006.

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 que rodam em um navegador Web, substituindo as ferramentas desktop. A cada dia surge uma nova ferramenta trabalhando em cima de ambiente 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 mundo do 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

Title: AN AJAX BASED REPORT MANAGER TOOL.

Currently, the corporative world characterized by the high level of competitiveness, ask for agility in the information retrieval process. To have the information is important, however, it couldn't be used as an isolated factor of success, in order to have abilities to perform operations better than your competitors. Considering that most companies already use relational databases management systems, the prominence could be, the way that you extract these informations. In this context, the use of tools to manage reporting services and tasks is growing, however, these tools demands technical knowledge of the users.New technologies are appearing, changing the way of developing Web Applications. These facts, are making possible to migrate tools and utilities from the desktop to the Web platform. Every day, a new tool appears,for example: text processors, electronic spreadsheets, personal scheduler, e-mail clients, and many others.So, this work, intend to present the innovations that appeared in the WWW world, the difficulties found in the current information systems, as well. After that, a proposal of development of a report manager tool under Web platform is made, in order to make such tasks more dynamic, flexible. Providing a user friendly interface, that enables the use by non technical users.

Key words: Information Retrieval, Reporting, Internet, AJAX.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AJAX	Asynchronous Javascript And XML
API	Application Program Interface
BI	Business Intelligence
CSS	Cascade Style Sheets
CSV	Comma Separated Value
DWR	Direct Web Remoting
ERP	Enterprise Resource Planning
FTP	File Transfer Protocol
IDE	Integrated Development Environment
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
NLS	oNLine System
PDA	Personal Digital Assistence
RIA	Rich Internet Application
SGBD	Sistema de Gerenciamento de Banco de Dados
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SSH	Secure Shell
TXT	TeXT
URL	Universal Resource Locator
XHTML	eXtensible Hypertext Markup Language
XML	eXtensible Markup Language
W3C	World Wide Web Consortium

SUMÁRIO

INTRODUÇÃO

No 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 a 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.

Conforme Silberschartz, Korth e Sudarshan (1999, p.1) "[...]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 (B*usiness 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].

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

[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.

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

Nesse trabalho de conclusão I, será feito um aparado geral sobre a evolução das aplicações disponibilizadas sob ambiente Web e sobre a importância dos relatórios para as organizações. Também serão descritas funcionalidades básicas da ferramenta. Porém, a implementação da solução será realizada durante o trabalho de conclusão II.

[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.

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 a ferramenta proposta será descrita sob linhas gerais, 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 chegarmos à Web atual, caracterizada por um cenário, composto de aplicações que até pouco tempo eram possíveis apenas em ambiente *desktop*.

1.1 Conceitos

Para a compreensão corretamente correta da evolução no mundo da internet, mais especificamente a World Wide Web, é fundamental conceituar alguns termos freqüentemente usados e muitas vezes confundidos em um primeiro momento.

Primeiramente Web, World Wide Web, WWW e W3 são termos equivalentes e não podem ser confundidos com a Internet. Enquanto a Internet é a grande rede, composta por uma infinidade de serviços e sistemas com propósitos diferentes. A Web é um destes sistemas, caracterizado por aplicações que foram criadas com o propósito de trocar informações por documentos de hipertexto[3].

> 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)

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

A Internet é uma rede constituída por um conjunto de outras redes, que se comunicam e colaboram entre si. Além de aplicações Web, podem ser encontrados serviços que viabilizam acesso remoto a máquinas, como por exemplo, Telnet e SSH; transferência de arquivos através do protocolo FTP; troca de mensagens, utilizando mensageiros instantâneos, como ICQ[4], MSN[5], googleTalk[6], Jabber[7]; troca de emails utilizando o protocolo SMTP, entre outros.

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 conhecemos como hipertexto, o lançamento do primeiro satélite artificial pela União Soviética, entre outros.

1.2.1 Os princípios do hipertexto

O Memex, codinome para *Memory Extension* (extensão da memória) pode ser encarado como o precursor do hipertexto. Criado durante a década de 30, tratava-se de um dispositivo mecânico capaz de criar e seguir associações, que permitia o armazenamento de uma quantidade grande de informações e rápida localização da mesma, simulando o funcionamento da mente humana, que trabalha por associação. Este é referenciado pelo seu criador Vannevar Bush, no artigo "*As We May Think*"[8] publicado em 1945, onde destaca-se, dentre outras coisas, o fato do aumento da soma dos conhecimentos ser alto e não acompanhado pelo aumento da capacidade de armazenamento e facilidade de acesso a informação.

> 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)

[4] Maiores informações em: http://www.icq.com
[5] Maiores informações em: http://imagine-msn.com/messenger/default2.aspx?locale=pt-br
[6] Maiores informações em: http://www.google.com/talk/
[7] Maiores informações em: http://www.jabber.org/
[8] O artigo está disponível na Internet no endereço: http://www.theatlantic.com/doc/194507/bush

Conforme Wikipedia (2006b), não há relação direta entre o hipertexto e o Memex, mas os estudos de Bush serviram como embasamento para a criação de outros sistemas, por exemplo, as enciclopédias eletrônicas, que trabalham sobre o mesmo conceito, isto é, a partir de uma informação pode-se criar associação com outra qualquer.

A evolução dos sistemas computacionais e a facilidade nas interfaces com o usuário são conseqüência do trabalho de muitas pessoas que não encaravam os computadores como super máquinas de fazer cálculos, mas sim como dispositivos que aliados ao conhecimento humano poderiam criar soluções com capacidade de processamento inimaginável por muitos.

Dentre essas pessoas, Joseph Carl Robnett Licklider, citado por muitos como o pai da Inteligência Artificial, defendia que a única maneira de se obter a completa potencialidade de um sistema computacional, era através de uma interface mais fácil entre usuários e computadores. No seu livro "*Man-Computer Symbiosis*"[9], publicado nos anos 60 ele expõe a idéia de que homens e computadores deveriam cooperar entre si na tomada de decisões, onde o homem estabeleceria objetivos, critérios e avaliaria os resultados e os computadores fariam o trabalho rotineiro.

Joseph afirmava que a combinação da inteligência humana com a capacidade de processamento dos computadores originaria sistemas capazes de processar mais informações do que qualquer cérebro humano poderia processar. Ao contrário de alguns pensadores da Inteligência Artificial, não era extremista a ponto de acreditar que uma máquina seria capaz de substituir um humano integralmente.

Muitos dos estudos citados anteriormente não foram contribuições práticas para o surgimento da *World Wide Web*, mas foram de fundamental importância para os criadores, como será mostrado em seguida.

O termo hipertexto foi utilizado a primeira vez por Theodor Holm Nelson, no seu projeto de mestrado em sociologia na universidade de Harvard. Este, consistia em criar um sistema semelhante a um processador de texto que viabilizasse a criação de diferentes versões de documentos e permitisse que essas versões e outros textos estivessem ligados de alguma forma. O projeto não foi concluído, mas Nelson continuou com seus estudos e em 1965 enviou um artigo a ACM onde cunhou o termo hipertexto.

[9] Uma versão online do livro *Man-Computer Symbiosis* está disponível em http://memex.org/licklider.pdf

Por definição no seu próprio nome, a World Wide Web define-se como uma teia, teia do tamanho do mundo, seria a tradução literal para o português, conforme é relatado em WIKIPEDIA (2006c,p.1).

Como já mencionado, as origens do hipertexto são vinculadas aos estudos de alguns cientistas, matemáticos e engenheiros americanos no período pós 2ª Guerra Mundial. Vannevar Bush, Ted Nelson e Douglas Engelbart merecem destaque pelo empenho durante décadas para solucionar o problema de sobrecarga de informações. Após a invenção do revolucionário Memex, Ted Nelson dá início ao projeto Xanadu[10], valendo-se das idéias de Bush, seguindo as mesmas idéias Engerlbart (o criador do mouse) criou e materializou o NLS[11] (oNLine System), conforme destaca Bello (2002,p.2).

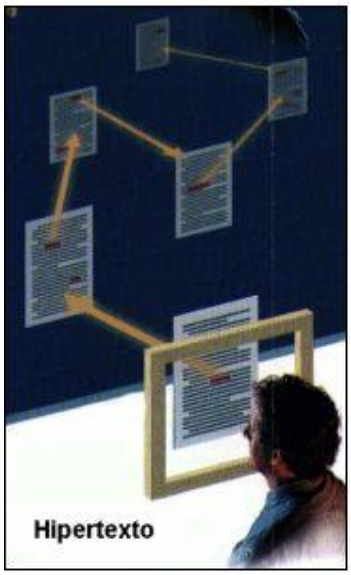

Figura 1 - Representação gráfica da interligação de documentos
Fonte: http://www.estudar.org/pessoa/internet/02www/people-tim_berners_lee.html

A Figura 1 representa graficamente a capacidade de referenciar outros documentos através do uso do hipertexto. O uso de hipertexto, faz parte da vida dos usuários de computador, alguns exemplos que merecem destaque são: ajuda online dos sistemas, sites, enciclopédias disponíveis em CDROM, entre outros.

[10] Sistema que permitia a interconexão entre todos os documentos registrados. Maiores informações podem ser obtidas em: http://xanadu.com/
[11] Sistema colaborativo revolucionário, foi o primeiro a utilizar o hipertexto na prática. Maiores informações podem ser obtidas em: http://en.wikipedia.org/wiki/NLS_%28computer_system%29

1.2.2 Tim Berners-Lee e 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*[12]**.** O ENQUIRE era um protótipo com algumas das principais funcionalidades da Web e principalmente dos WIKIS[13], muito utilizados nos dias de hoje, e destacava-se pelas seguintes características:

- Banco de dados, todos os dados armazenados poderiam ser alterados a qualquer momento;
- Criação de relacionamentos, hiperlinks bidirecionais;
- Edição direta no servidor;

Por se tratar de um centro de pesquisa, onde um dos objetivos é o compartilhamento de informações entre os cientistas, e no caso do CERN, muitos deles estavam fora do instituto e até mesmo fora do país, enfrentavam sérios problemas para viabilizar a troca de conhecimento, dificultando o trabalho.

Os problemas encontrados não paravam por aí e iam muito além da ausência de um meio de documentação *Online*. Assim, os documentos estavam distribuídos de maneira descentralizada, partindo da premissa que, muitos deles em sua versão inicial eram apenas um esboço e cresciam rapidamente, trazendo a necessidade de redistribuir o documento alterado para todos os envolvidos ou interessados no projeto. Além disso, o alto nível de rotatividade de pessoal, também gerava dificuldade para descobrir onde estavam os documentos atualizados. Alem desses fatores, a questão geográfica acrescentava dificuldades de encontrar a solução adequada para viabilizar a troca de informações entre todos os cientistas localizados em lugares diferentes do continente europeu.

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.

---

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

[13] 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

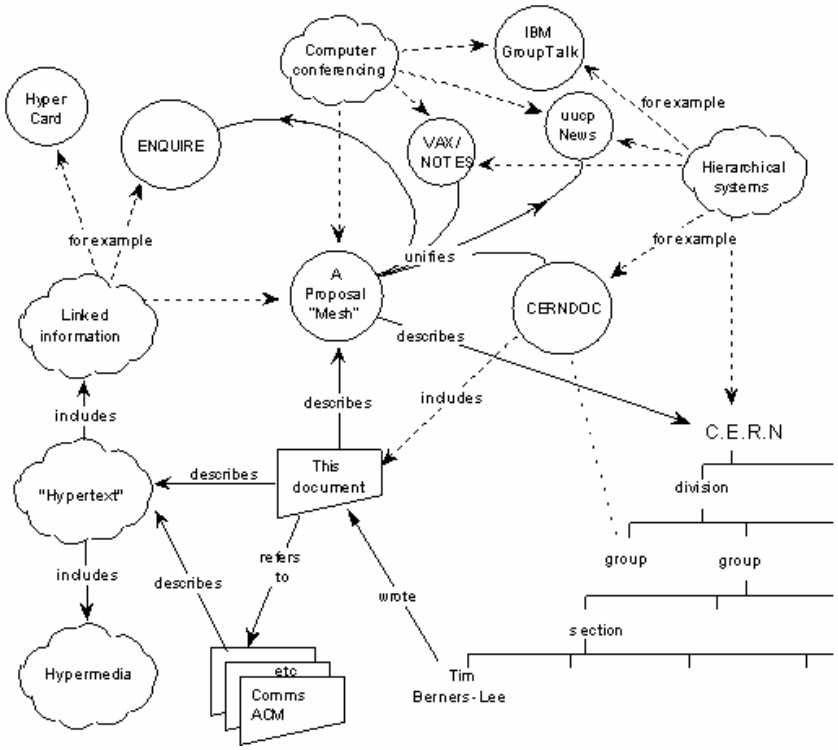

Figura 2 - Representação gráfica da proposta de Tim Berners-Lee
Fonte: http://www.w3.org/History/1989/Image1.gif

A Figura 5 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. Abaixo, encontra-se uma representação gráfica da solução proposta:

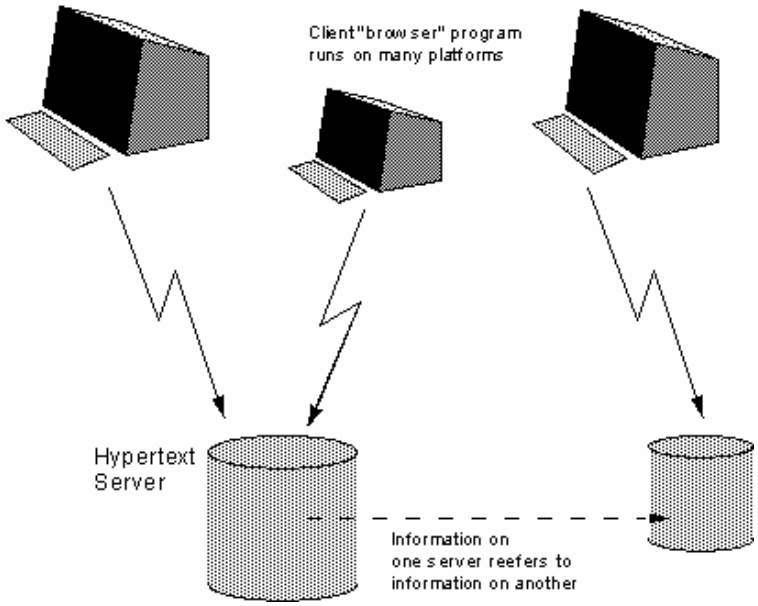

Figura 3 – Arquitetura que viabiliza a comunicação entre diferentes plataformas
Fonte: http://www.w3.org/History/1989/Image2.gif

Paralelamente aos fatos ocorridos no CERN, estudos sobre um protocolo que viabilizasse a organização de forma hierárquica da informação estavam surgindo, o *Gopher*.

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 – ta-da! – 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.

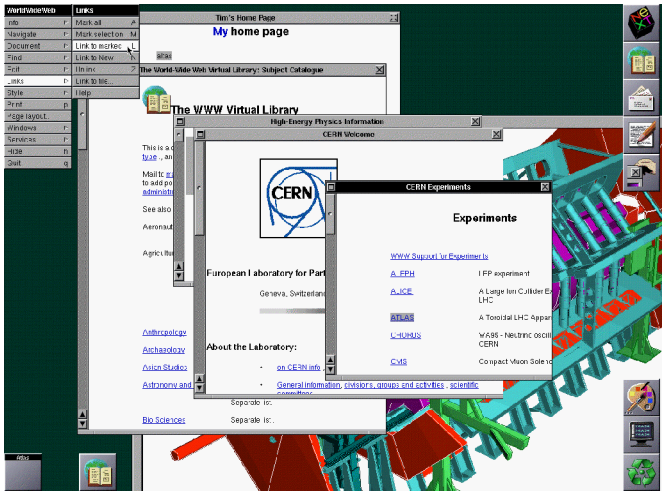

Figura 4 - O WorldWideWeb rodando na plataforma NEXT
Fonte: http://en.wikipedia.org/wiki/WorldWideWeb

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 mais compatível 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.

1.2.3 A World Wide Web da Europa para o Mundo

Até então tudo que se tinha em relação a World Wide Web, o navegador, o servidor Web estavam na Europa, mais precisamente no CERN, onde tudo começou. O primeiro servidor Web nos EUA entrou em atividade em dezembro de 1991, novamente em um centro de pesquisa, o SLAC (*Stanford Linear Accelerator Center*), na Califórnia.

Na 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.

Ficava cada vez mais claro que a equipe no CERN não conseguiria levar o projeto adiante, então Tim resolveu fazer um apelo a outros desenvolvedores para juntarem-se ao projeto. Alguns projetos surgiram individualmente, porém sem muito sucesso, foram eles:

- MIDAS
- Viola
- Erwise

Mesmo sem muito sucesso, os projetos individuais continuaram, surgiu o MOSAIC, no NCSA (*Nacional Center for Supercomputing Applications*) desenvolvido sob um ambiente *X Window*, muito conhecido na comunidade, permitindo uma interface com o usuário baseada em janelas. Esse navegador foi lançado mais tarde para as plataformas PC e Mac, o que ocasionou um grande crescimento na utilização da Web. O sucesso do MOSAIC levou seu criador, Marc Andreesen e mais alguns estudantes a sair do NCSA e fundar a Mosaic Communications Corporation que mais tarde originou a Netscape, conhecida pelo seu navegador Netscape Navigator.

O crescimento era notório, o tráfego de informações gerado pela Web crescia 341,634% anualmente. (ZAKON,2005,p11)

1.2.4 1994, o ano 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 (*Massachussets Institute of Technology*) 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 As principais iniciativas foram as ações visando estabelecer regras e padrões documentados, seguindo 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, muito falado nos dias de hoje, é tido como uma nova geração de aplicativos, também chamado de SOA, *Service Oriented Architecture*.

O W3C, 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*.

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.

Muitos problemas oriundos da proposta inicial da World Wide Web dificultam as coisas, tanto pelo modo de conexão, como a maneira de que os dados trafegam repetidamente.

A troca de mensagens entre o cliente e o servidor faz com que a interação seja lenta, pois a cada requisição do cliente é esperado o retorno do servidor para que então a próxima requisição seja feita. 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.

A solução proposta anteriormente funciona, mas oferece alguns pontos negativos:

- É pouco elegante;
- Gera um *overhead*[14] na página web, desnecessário com objetos ocultos, responsáveis pela comunicação;
- Não é compatível com os padrões estabelecidos pelo W3C;

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, é acessível através de JavaScript, VBScript e outras linguagens de *script* suportadas pelos navegadores.

Originalmente implementado no navegador da Microsoft, o Internet Explorer 5.0, com o nome XMLHTTP, através de um controle ActiveX[15]. Em 2002, foi implementado o suporte nativo com a implementação do XMLHTTPRequest ao navegador Mozilla 1.0, essa implementação foi seguida pelos demais navegadores, utilizados atualmente, como nos retrata Wikipedia (2006,p.2):

- Safari 1.2, da Apple;
- Konqueror;
- Opera 8.0, da Opera Software;

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.

Conforme W3C (2006c, p.1):

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

> 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 difusão e adoção do uso do XMLHttpRequest, contribui e pode ser considerado um dos componentes mais importantes do AJAX.

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 por "debaixo do pano", 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.

Como é possível concluir, não é uma nova tecnologia, AJAX costuma ser definido como um conjunto de tecnologias que foram agrupadas e cooperam entre si para buscar um fim específico, segundo Garrett(2005,p.1), dentre as tecnologias envolvidas, tem-se:

- Visualização baseada em *Web Standards*[16],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.

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

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

2 DESMISTIFICANDO O 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, 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 oferecia tecnologias muito limitadas, trazendo a necessidade de usar maneiras alternativas para se obter uma interface um pouco mais rica, o esforço não compensava, como foi visto anteriormente umas das possibilidades era a utilização de elementos ocultos, IFRAMES, que auxiliavam no fluxo da aplicação, porém essas soluções não eram muito atrativas. 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 dessa seção é apresentar as vantagens e considerações no uso da tecnologia AJAX para o desenvolvimento de aplicações Web ricas, chamadas de RIA[17].

[17] 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:

1. O usuário fazia uma requisição;
2. O servidor fazia algum processamento, interagindo com sistemas legados;
3. O HTML então era devolvido ao cliente, mostrando o resultado do processamento para o usuário.

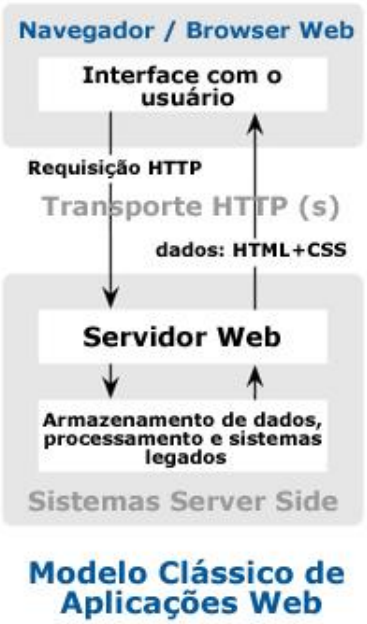

Figura 5 - Modelo de Arquitetura Clássica de Aplicações Web
Imagem do autor, adaptado de GARRETT (2005)

A Figura 5 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 veremos 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 temos 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 6 apresenta como ficaria a arquitetura da mesma aplicação utilizando AJAX:

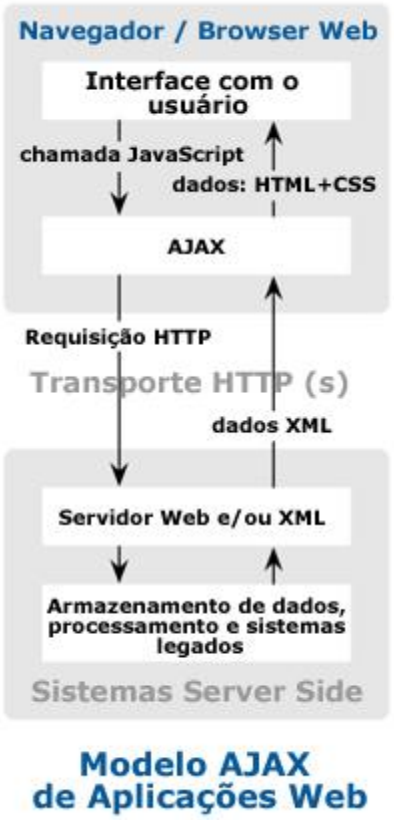

Figura 6 - Modelo de Arquitetura de Aplicações Web utilizando AJAX
Imagem do autor, adaptado de GARRETT (2005)

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[18] um objeto XMLHttpRequest. Existem maneiras diferentes de se fazer isso nos navegadores atuais, visto que, o Internet Explorer não suporta

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

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 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

Fonte: Do autor

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*[19], 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);
      }
}
```

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

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

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 (*Autocompletion*) – 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 pagina 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*[20].

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 - Diferenças entre os aspectos técnicos e gerenciais

Desenvolvedores falam sobre	Gestores falam sobre
Usabilidade	Eficiência
Largura de banda	Custo de transações
Ramp-up[21]	Custos com treinamento
TCO[22]	ROI (Retorno sobre o investimento)

Fonte: Do autor

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.

[20] Processo que roda paralelamente ao fluxo normal da aplicação, permanece executando sem interromper a aplicação principal.
[21] Capacidade de aumentar a produtividade de uma empresa, antecipando-se a uma possível demanda, agir antecipadamente, ação pró-ativa.
[22] Índice financeiro para mensurar custos diretos e indiretos na compra de algo.

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 7 - Comparativo do tempo consumido
Imagem do autor, adaptado de JENSEN(2006,p.3)

Como é mostrada na Figura 7 a implementação utilizando a arquitetura AJAX permitiu que o processo escolhido fosse executado em menos tempo que a arquitetura tradicional.

Figura 8 - 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 Contras

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*[23] de aplicações existentes.

Essa seção fará um aparato 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:

- Os desenvolvidos com o objetivo de facilitar a integração com alguma tecnologia *server side*[24], 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 abaixo 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 aplicacoes Web dinamicas, 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.

[23] 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.
[24] Script ou conjunto de instruções que são processadas no servidor e o resultado do processamento pode ser usado para gerar saída para o cliente.

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 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*[25], 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 nosso cotidiano 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,

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

Flicker entre outras inúmeras ferramentas que estão substituindo aplicações desktop e trazendo a funcionalidade para um navegador Web.

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).

Como nos retrata 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*[26](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.

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,p.), obtidas através da experiência acumulada por empresas fornecedoras em repetidos processos de implantação.

[26] 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

Davenport (apud BERVIAN,2000,p.), 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 acima, 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[27], BusinessObjects[28], entre outras. A crescente evolução no uso de ferramentas de BI, conseqüentemente de *reporting*[29], também despertou a atenção da comunidade do software livre. Abaixo são listadas algumas soluções.

BIRT[30], 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).

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;

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

- 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 nos retrata 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*[31] vem crescendo, e está se tornando uma ferramenta importante para auxiliar no processo de tomada de decisão das empresas.

[31] 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 estrutura da ferramenta a ser concebida: um gerenciador de relatórios rodando sob a plataforma Web.

4.1 Acessibilidade e interface com o usuário

Para melhor atender as demandas do mundo moderno, a ferramenta será concebida em forma de aplicação Web, visando alta disponibilidade e permitindo o acesso a partir de qualquer lugar do mundo através de um navegador / browser.

A interface será concebida utilizando XHTML e CSS, pois evita a necessidade de instalação e configuração de plugins ou extensões como por exemplo, o uso de Flash ou Applets JAVA. Optou-se por utilizar XHTML, pois é uma linguagem de marcação baseada em XML, 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 é uma das premissas defendidas pelos padrões e definições da Web semântica. Esse nome é dado, pois ao invés de ser apenas uma linguagem de marcação, a estrutura do XHTML, não traz apenas informações ou conteúdo, mas também dados semânticos para o dispositivo que está visualizando o documento. Já o uso de CSS, além de trazer facilidade na alteração do layout, é utilizando com o objetivo de seguir a

metodologia *tableless*[32], de modo a oferecer páginas Web mais leves e semanticamente corretas.

Outro ponto forte que se sobressai nas características da interface com o usuário, é a necessidade de interação de maneira rápida e intuitiva. Para se obter esse elevado padrão de qualidade, será utilizada a tecnologia AJAX, que permitirá uma interação mais dinâmica entre a aplicação e o usuário. Fazendo com que ele se sinta a vontade em utilizar o sistema, sem precisar esperar os longos intervalos de tempo entre as interações (característica marcante e indesejada, para quem usa a Web, como apresentado anteriormente).

As telas buscarão ao máximo seguir o padrão utilizado por aplicações desktop, de modo a trazer uma maior familiaridade do usuário com a ferramenta.

4.2 Independência de banco de dados

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

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

O suporte a múltiplas fontes de dados, dentre elas: *result sets*, arquivos XML, arquivos TXT, arquivos CSV, será possível através de um processo de importação e transformação em uma estrutura auxiliar que será interpretada pela ferramenta para a geração do relatório, viabilizando a integração com fontes de dados que não estão 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. A partir da exportação dos dados em um dos formatos suportados, seria possível trabalhar com os dados, até então inacessíveis pela ferramenta.

[32] 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

4.3 Funcionalidades

Após obter os dados de alguma das possíveis fontes, a ferramenta permitirá realizar consultas e extrair informações. Além de permitir a navegação nas informações, será possível formular e formatar um relatório, que poderá ser impresso ou exportado em formatos específicos para viabilizar a integração com outras ferramentas.

Ao realizar a conexão com um banco de dados, todas as entidades que podem retornar dados são apresentadas para os usuários: tabelas, *views*[33] e *stored procedures*[34]. A partir da visualização das entidades que podem fornecer dados, o usuário poderá montar, um *data set*, que nada mais é que um conjunto de informações a serem apresentadas em forma de relatório. Filtros poderão ser aplicados, sem conhecimento algum de linguagens de consulta, com o objetivo de fornecer uma alternativa para pré-filtrar informações, evitando processamento desnecessário em informações que não são importantes sob um determinado contexto. Esse filtro poderá ser feito, de duas formas, trabalhando com as informações propriamente ditas, onde o filtro será aplicado sobre o valor, ou filtrando quais dados deverão ser considerados de uma determinada fonte. Por exemplo, a seleção de quais colunas serão disponibilizadas para que o usuário "monte" o relatório.

Além de permitir a exibição de informações que vêm prontas do banco de dados, também será possível efetuar operações matemáticas, por exemplo, um totalizador; ou ainda formatação condicional, dependendo se uma determinada condição é satisfeita. O valor é exibido de forma diferente, permitindo destacar informações, que podem estar num estado crítico, por exemplo, o valor mínimo de produtos em um estoque. Além disso, strings de controle (mascaras) poderão ser aplicadas ao valor de uma coluna, permitindo formatar o valor. Por exemplo, a exibição de um campo de CPF, pode ser armazenado sem os caracteres especiais que o formam, mas ao exibir a informação com a máscara (caracteres especiais de formatação) , fica mais claro e visível, facilitando a interpretação do mesmo.

Além das funcionalidades frequentemente utilizadas em um gerenciador de relatórios, a ferramenta proposta oferecerá recursos que viabilizarão processos em *batch* de extração de informações. Os *data sets*, bem como os *layouts* dos relatórios poderão ser salvos, permitindo que se crie modelos de relatórios, facilmente reutilizáveis. Sendo assim, será possível agendar

[33] São estruturas semelhantes a tabelas, porém são construídas a partir de consulta de dados.
[34] Coleção de comandos em SQL organizados de forma lógica, como um mini programa que fica armazanado no banco de dados.

a execução de um relatório, que poderá ser enviado por e-mail ou exportado em um arquivo, permitindo que a informação já esteja disponível em um determinado momento, por exemplo, o resumo de transações efetuadas no dia anterior para o diretor de uma empresa de transações.

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 proposta busca atender as necessidades de usuários não integrantes da equipe de desenvolvimento do software.

Na primeira etapa desse trabalho, o objetivo principal foi apresentar o problema que a ferramenta que será desenvolvida se propõe a resolver, bem como as tecnologias que serão utilizadas para realizar o fim proposto. O uso de AJAX na concepção da ferramenta viabiliza o desenvolvimento da mesma, uma vez que, um dos pontos fortes de tal ferramenta é a interação com o usuário.

Os frameworks que apenas foram citados nessa primeira etapa serão cuidadosamente avaliados com o objetivo de verificar a conformidade e a contribuição para o desenvolvimento da ferramenta proposta.

Já o segundo momento desse trabalho, fica como objetivo principal a implementação da ferramenta que será capaz de solucionar os problemas aqui apresentados.

REFERÊNCIAS BIBLIOGRÁFICAS

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 09/02/2005.

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.

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.

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.

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.

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.

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