CENTRO UNIVERSITÁRIO FEEVALE

IGOR HENRIQUE BERLITZ

GERADOR GRÁFICO DE RELATÓRIOS UTILIZANDO A CLASSE FPDF

Novo Hamburgo, dezembro de 2007.

IGOR HENRIQUE BERLITZ

GERADOR GRÁFICO DE RELATÓRIOS UTILIZANDO A CLASSE FPDF

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, dezembro de 2007.

AGRADECIMENTOS

Um agradecimento especial aos meus familiares que sempre estiveram ao meu lado, apoiando, incentivando, compreendendo e acreditando nos meus estudos.

Agradeço a todos os professores que de uma forma ou outra contribuíram para a minha formação.

Ao meu orientador Juliano Varella de Carvalho, pela atenção, dedicação, incentivo e fundamental apoio nas sugestões e revisões deste trabalho.

RESUMO

Atualmente vivemos em um mercado globalizado e competitivo, caracterizado por constantes transformações, onde cada vez mais se propaga a idéia de que uma empresa é essencialmente uma processadora de informações. A tecnologia da informação (TI), que durante muito tempo foi considerada um item de suporte administrativo, passa a ser encarada como um recurso facilitador na tomada de decisões gerenciais seja através da disponibilização das informações adequadas, indicação de novas tendências de mercado ou simplesmente auxílio à localização e compartilhamento do conhecimento. Com o amadurecimento da Internet, e conseqüentemente o crescimento das aplicações WEB, o número de informações e relatórios disponibilizados pelas organizações tem tomado proporções gigantescas. Neste contexto, uma grande dificuldade encontrada pelos desenvolvedores é proporcionar aos usuários a impressão de seus relatórios na WEB de forma correta e ordenada. Uma forma de resolver esse problema é possibilitar a geração dos relatórios no formato PDF, pois esse tipo de arquivo oferece saída consistente e de alta qualidade, permitindo a geração de documentos imprimíveis e atraentes. A evolução das tecnologias WEB tem permitido que esse poderoso recurso, esteja disponível em praticamente todas as linguagens de desenvolvimento para a Internet. Entretanto, as bibliotecas e classes existentes para geração destes relatórios exigem dos desenvolvedores um trabalho cansativo, pouco produtivo e de difícil adequação às alterações no layout dos relatórios. Por isto, este projeto propõe uma solução a ser desenvolvida utilizando os principais conceitos da WEB 2.0, a tecnologia AJAX e a classe FPDF (escrita em PHP), com o intuito de proporcionar aos desenvolvedores um ambiente de fácil manipulação para geração de relatórios no formato PDF.

Palavras-chave: Gerador de relatórios, AJAX, FPDF, Sistemas WEB, WEB 2.0.

ABSTRACT

Nowadays we live in a globalized and competitive world, characterized by the constant transformations, where it propagates the idea that a company is essentially a information processing. The technology of the information (TI), that during a long time was considered an item of administrative help, pass to be faced as assistant resource to take management decisions or through the arrangement of appropriate information, or indication of new market trends or simply aid the searching and sharing of knowledge. Through the internet's growth, and consequently the growth of WEB applications, the number of information and reports supplied by the organizations has taken huge ratio. At this context, the difficulty most common found by the developers is to provide to the users a way to print theirs reports on the WEB in a correct and commanded form. A solution to this problem is to make possible to generate these reports on PDF format, because this type of file provides consistent output and high quality, allowing the generation of attractive and printable documents. The evolution of the WEB technologies have allowed that this powerful resource be available on practically all web development language. However, the libraries and class existing for the generation of these reports demand of developers a tiring work, little productive and hard adequacy to the alterations on the report layout. For this, the project set out to study a solution to be developed using the main concepts of WEB 2.0, the AJAX technology and the FPDF class (wrote in PHP), with the purpose to provide to the developers an environment of easy manipulation for the generation of PDF reports.

Key words: Report Generator, AJAX, FPDF, WEB Systems, WEB 2.0.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ADO	ActiveX Data Objects
AJAX	Asyncronous Javascript and XML
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASP	Active Server Pages
BDE	Borland Database Engine
BI	Business Intelligence
BIRT	Bussiness Intelligence and Reporting Tools Project
BMP	BitMap Picture
CCITT	Consultative Committee for International Telephone and Telegraph
CFML	ColdFusion Markup Language
CSS	Cascading Style Sheets
CSV	Comma Separated Value
DHTML	Dynamic HTML
DLL	Dynamic Link Library
DOM	Document Object Model
DWR	Direct WEB Remoting
GED	Gerenciamento Eletrônico de Documentos
GIF	Graphics Interchange Format
GPL	General Public License
GWT	Google Web Toolkit
HTML	Hypertext Markup Language
HTTP	HyperText Transfer Protocol
IBX	InterbaseExpress
IIS	Internet Information Server

IPS	Interchange PostScript
J2EE	Java 2 Enterprise Edition
JBIG	Joint Bi-Lebel Image Experts Group
JPEG	Joint Photographic Experts Group
JSP	JavaServer Pages
LZW	Lempel-Ziv-Welch
MVC	Model View Controller
NDR	Rave Reports Report File
ODBC	Open Data Base Connectivity
PC	Personal Computer
PDF	Portable Document Format
PDL	Page Description Language
PHP	Hypertext Preprocessor
PNG	Portable Network Graphics
PWS	Personal Web Server
RDBMS	Relational Database Management System
RLE	Run Length Encoding
RSS	Really Simple Syndication
RTF	Rich Text Format
SGBD	Sistema de Gerenciamento de Banco de Dados Relacional
SGML	Standard Generalized Markup Language
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
TI	Tecnologia da Informação
TIFF	Tagged Image File Format
UDDI	Universal Description and Discovery of Information
XHTML	Extensible Hypertext Markup Language
XLS	Excel File Format
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformation
W3C	World Wide Web Consortium
WSDL	Web Services Description Language

SUMÁRIO

INTRODUÇÃO

Nunca a informação foi tão importante para a nossa economia (FERAUD apud DAVENPORT, DICKSON e MARCHAND, 2004). Em um mercado globalizado e competitivo, que se transforma a todo instante, em uma velocidade alucinante, as organizações vêem-se obrigadas a possuir um sistema de informação ágil que acompanhe essas transformações.

Neste cenário competitivo onde a resposta às mudanças deve ocorrer de maneira rápida e eficaz, as organizações passam a valorizar a informação como um recurso estratégico. Cada vez mais se propaga a idéia de que uma empresa é essencialmente uma processadora de informações.

> A informação deve ser considerada como diferencial de negócios quando proporciona alternativas de lucratividade e retornos profícuos para a empresa, seja sedimentando atuações, implementando os atuais negócios, seja ainda, criando novas oportunidades de negócios (REZENDE e ABREU, 2000, p.65).

O papel da tecnologia da informação (TI) mudou significamente nos últimos anos. Com a globalização e aumento da concorrência, a TI que durante muito tempo foi considerada apenas um item de suporte administrativo, passa a ser encarada como um importante recurso, que agrega valor ao produto final ou serviço, resultando no crescimento dos lucros e redução dos custos operacionais.

O surgimento de inovações em termos de TI cresceu substancialmente nos últimos anos, com destaque para a disseminação de redes, especialmente a Internet, que se constitui atualmente no conjunto de TI com maior impacto na sociedade. Conforme Laudon e Laudon (2001), devido a sua velocidade de crescimento e novas formas de comunicação, a Internet está possibilitando mudanças importantes na forma de conceber e de realizar os negócios e atividades nas organizações.

De acordo com O'Brien (2006) em poucos anos a Internet se estabeleceu como uma plataforma tecnológica livre de muitas fronteiras e limites internacionais convencionais. Com esse amadurecimento da Internet, as corporações perceberam que ela pode ser usada como um meio de prestação de serviços e não apenas como publicação de conteúdo em sites com o objetivo de disseminar informações. Dessa forma, os sites deixam de ter uma característica estática para se tornarem verdadeiros aplicativos no servidor.

Em decorrência do crescimento das aplicações WEB, o número de informações e relatórios disponibilizados pelas organizações tem tomado proporções gigantescas. Muitas soluções desenvolvidas, ainda que com ótimo grau de utilidade e benefício, deixam a desejar no momento que disponibilizam documentos em formatos de difícil utilização, ou então, quando seus relatórios são bem visualizados no *browser*, mas não oferecem uma impressão adequada.

Nesse contexto, a grande dificuldade encontrada pelos desenvolvedores é possibilitar aos usuários a impressão dos seus relatórios de forma correta e ordenada. Desenvolver para a WEB, passa por adotar padrões, como HTML (*HyperText Markup Language*) e Javascript, que imperam como tecnologia no lado cliente. O *browser* por sua vez, disponibiliza o ambiente de interação entre o usuário e a aplicação, e dessa forma tem total controle sobre a impressão. Assim, percebe-se que existe um problema na falta de suporte da linguagem utilizada, muitas vezes HTML, pois é ela que fornece a saída ao conteúdo destinado à impressão via *browser*.

Segundo Welling e Thomson (2003) uma forma de resolver esse problema é possibilitar a geração dos relatórios no formato de arquivo PDF (*Portable Document Format*), pois esse tipo de arquivo oferece saída consistente e de alta qualidade, extremamente útil na distribuição de documentos imprimíveis e atraentes, características de extrema importância em aplicações WEB.

A constante evolução das tecnologias WEB tem permitido que esse poderoso recurso, esteja disponível em praticamente todas as linguagens de desenvolvimento para a Internet, das quais podemos incluir PHP (*Hypertext Preprocessor*), ASP (*Active Server Pages*), ColdFusion e JSP (*JavaServer Pages*).

O PHP é uma linguagem *script open source* de uso geral, muito utilizada e especialmente criada para o desenvolvimento de aplicações WEB (PHP, 2007). Cada vez

mais, as empresas vêem o PHP como alternativa apropriada no desenvolvimento de aplicações de pequeno e médio porte para a Internet.

Conforme Welling e Thomson (2003) o PHP disponibiliza suporte nativo para a criação de documentos PDF através de duas bibliotecas, PDFlib e a ClibPDF, que por contarem com bibliotecas externas, não estão compiladas para o PHP por padrão. Essas duas bibliotecas são semelhantes, pois fornecem uma API[1] (*Application Programming Interface*) de funções que geram os documentos no formato PDF. Podem ser utilizadas sem custo para uso pessoal, mas exigem o pagamento de uma taxa de licença se forem utilizadas em aplicações comerciais.

Fuecks (2003) explica que outra forma de geração de documentos PDF com PHP é a utilização das classes R&OS PDF Class e FPDF. Diferentemente da PDFlib e da ClibPDF, elas não são bibliotecas propriamente ditas e sim classes escritas em PHP. Essas classes são gratuitas, portanto para utilizá-las basta fazer uma cópia dos seus arquivos no servidor WEB e referenciá-las nos programas que desejam fazer impressões em PDF.

Cabe ressaltar, que a classe FPDF apresenta algumas desvantagens que levaram ao estudo mais aprofundado da implementação de uma ferramenta de auxílio na geração de arquivos no formato PDF:

- Não possui uma interface de auxílio para o desenvolvimento automatizado de *layouts* de telas e relatórios a serem exportados;
- Necessita definir manualmente no código PHP a formatação e o posicionamento do texto e de todos os elementos que formam o relatório;
- Não permite o posicionamento de textos na diagonal da página.

Paralelamente a isso, uma nova geração de serviços disponíveis na Internet vem surgindo, a denominada WEB 2.0. Tim O'Reilly no artigo intitulado "*What is Web 2.0*" (O'REILLY, 2005), define a WEB 2.0 como uma nova geração de serviços de Internet

[1] API – do inglês *Application Programming Interface* (Interface de Programação de Aplicativos). Conjunto de rotinas e padrões estabelecidos por um *software* para utilização de suas funcionalidades por programas aplicativos.

baseada em aplicações mais rápidas, mais leves e com mais interação entre o usuário e os conteúdos disponíveis, utilizando tecnologias como AJAX, DOM e Script.aculo.us.

Sendo assim, a motivação deste trabalho é projetar e desenvolver uma ferramenta gráfica em PHP, para ser utilizada na WEB, que permitirá aos desenvolvedores de *software* a geração de relatórios no formato PDF, fazendo uso da classe FPDF já existente. A escolha da classe FPDF como base para o desenvolvimento dessa ferramenta tem como justificativas:

- Permite utilização gratuita tanto para uso pessoal quanto comercial;
- Classe de simples manuseio, que possui inúmeras funcionalidades a serem exploradas, seja de maneira direta ou através de suas extensões;
- Possui métodos que podem ser sobrescritos, permitindo a herança de classes e implementação de métodos próprios;
- Funciona em qualquer servidor PHP.

Com base em informações sobre a melhor forma de desenvolvimento de aplicações que rodam na Internet, e análise dos principais problemas encontrados na utilização da classe FPDF, o sistema proposto tem como foco principal fornecer recursos para a geração de relatórios PDF, através de uma interface WEB intuitiva, desenvolvida utilizando conceitos de usabilidade, permitindo uma fácil manipulação por parte dos desenvolvedores.

A estrutura deste trabalho está dividida em quatro capítulos. O primeiro capítulo apresenta os principais formatos de arquivos utilizados na geração de relatórios corporativos, algumas ferramentas de relatórios existentes no mercado, linguagens WEB que possuem suporte para a criação de documentos PDF, e por fim, bibliotecas e classes PHP que auxiliam na geração de relatórios no formato PDF.

No segundo capítulo é dada uma visão sobre a nova geração de aplicações WEB, onde serão apresentados conceitos relacionados à WEB 2.0 e AJAX. O terceiro capítulo mostra as dificuldades enfrentadas pelos desenvolvedores na geração de relatórios PDF ao necessitar manipular manualmente a classe FPDF. No último capitulo serão apresentadas as tecnologias utilizadas no desenvolvimento da aplicação, as principais telas da ferramenta e os detalhes de implementação de suas funcionalidades.

1 GERAÇÃO DE RELATÓRIOS

Com o amadurecimento da Internet, e conseqüentemente o crescimento das aplicações WEB, o número de informações e relatórios disponibilizados pelas organizações tem tomado proporções gigantescas. Muitas soluções desenvolvidas, ainda que com ótimo grau de utilidade e benefício, deixam a desejar no momento que disponibilizam documentos em formatos de difícil utilização, ou então, quando seus relatórios são bem visualizados no *browser*, mas não oferecem uma impressão adequada.

Diante deste cenário, o objetivo deste capítulo é apresentar os principais formatos de arquivos utilizados na geração de relatórios corporativos, ferramentas de relatórios existentes no mercado, linguagens WEB que possuem suporte para a criação de arquivos PDF e, por fim, bibliotecas e classes PHP que auxiliam na geração de documentos PDF.

1.1 PDF – *Portable Document Format*

Criado pela Adobe Systems, o formato PDF (*Portable Document Format*) é uma especificação disponível publicamente, usada por entidades de padronização no mundo inteiro para a distribuição e a troca mais seguras e confiáveis de documentos eletrônicos. Os arquivos PDF são compactos e completos, podendo ser compartilhados, visualizados e impressos por qualquer pessoa com o *software* gratuito Adobe Reader (ADOBE, 2007b).

1.1.1 Breve histórico

O formato PDF nasceu do entusiasmo de John Warnock – um dos fundadores da Adobe – sobre o conceito de documento eletrônico. Seu objetivo era criar uma maneira de distribuir documentos na empresa, de modo que pudessem ser visualizados em computadores com qualquer sistema operacional e impressos em qualquer impressora. Apesar de a meta

inicial ser a criação de um *software* para uso interno, Warnock já previa que essa tecnologia poderia ser adotada em outros tipos de aplicações, tais como, relatórios, enciclopédias, mapas militares, manuais e livros (GREGO, 2006).

Antes da criação do formato PDF a Adobe já possuía dois produtos que tinham alguma relação com documentos eletrônicos. O primeiro era a linguagem PostScript, usada para descrever páginas a serem impressas, e o outro era o Illustrator, um aplicativo para ilustrações baseado nessa linguagem. Esses dois *softwares* foram o ponto de partida para o desenvolvimento do padrão PDF, inicialmente chamado de IPS (*Interchange PostScript*).

Conforme Grego (2006) no segundo semestre de 1991, em um congresso nos Estados Unidos, a Adobe fez a primeira apresentação pública da tecnologia PDF, até então identificada com o nome provisório de IPS. No entanto, somente no final de 1992, foi disponibilizada oficialmente a versão 1.0 do padrão PDF, composta de dois programas: o Acrobat, uma ferramenta para a criação de documentos nesse padrão, e o Reader, *software* para visualização dos arquivos PDF.

Inicialmente, a tecnologia PDF não obteve o sucesso esperado por seus criadores. Na verdade, esse padrão demoraria anos para se tornar popular. E motivos não faltaram para isso. Apesar de já possibilitar a inclusão de *links*, marcadores e fontes de caracteres nos documentos eletrônicos, o padrão PDF sofria com a concorrência de outras opções conhecidas no mercado, como Envoy[2], Common Ground[3] e Djvu[4].

Quando foi lançada, a tecnologia PDF custava 695 dólares e não fazia muito mais do que transformar outros formatos de arquivo em PDF, algo que hoje pode ser feito com programas gratuitos, tornando-se assim uma solução cara. Na mesma época, uma edição do *software* para uso em redes valia em torno de 2500 dólares e o visualizador Reader custava 50 dólares. O formato PDF começou a ganhar uma aceitação significativa no mercado, no momento que a Adobe passou a distribuir gratuitamente o Reader em 1994 (ADOBE, 2007a).

---

[2] ENVOY. Maiores informações em: http://www4.envoytech.com/envoy/.
[3] COMMON GROUND. Maiores informações em: http://www.hummingbird.com.
[4] DJVU. Maiores informações em: http://djvu.org.

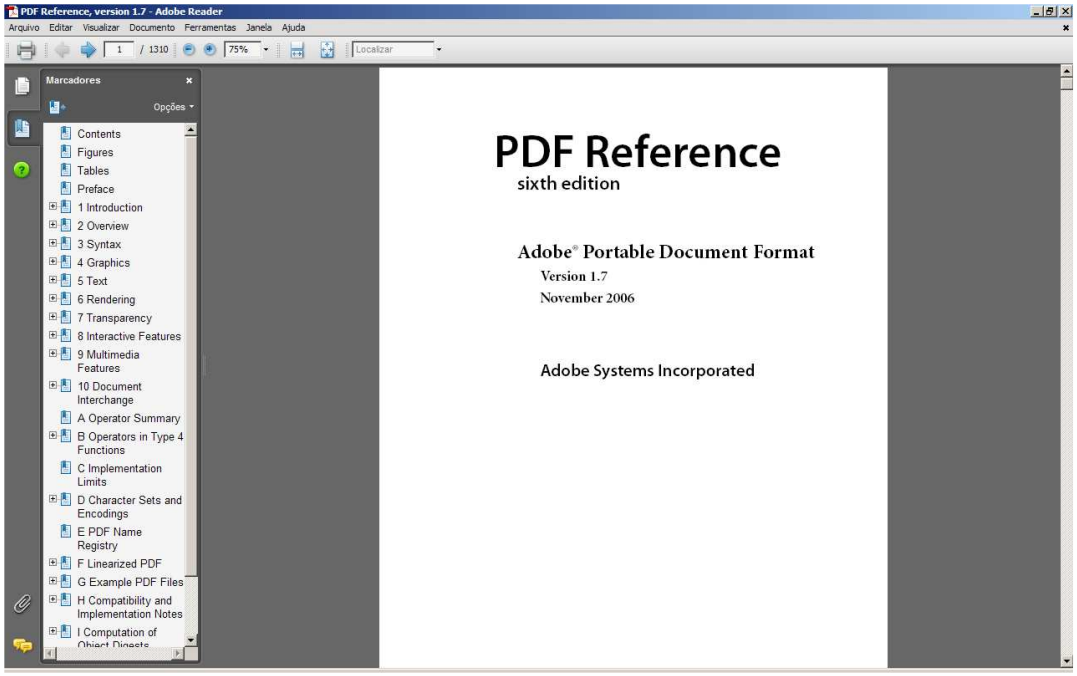


Figura 1.1 - Tela do visualizador Adobe Reader, atualmente na versão 8.0.

1.1.2 Características de um arquivo PDF

Nos últimos anos a documentação eletrônica tem assumido um papel importante em todas as áreas de negócios e produção. As empresas têm adotado o padrão PDF para otimizar e agilizar o seu fluxo de trabalho e documentação. Segundo Adobe (2006), o formato PDF possui características interessantes, dentre elas, destacam-se:

- **Portabilidade**: Representa um documento com fidelidade ao seu formato original independente do *software* de aplicação e versão, do *hardware* e do sistema operacional que foi usado para criá-lo;

- **Compressão**: O arquivo em formato PDF é compacto, pois possui diversas técnicas de compressão de dados. A tabela 1.1 apresenta os formatos de compressão utilizados pelo padrão PDF;

Tabela 1.1 - Formatos de compressão utilizados pelo padrão PDF.

Imagens Policromáticas e Monocromáticas	**Imagens Binárias**	**Texto**
JPEG	JBIG2	LZW
JPEG 2000	CCITT	Flate
	RLE	

Fonte: Adaptação do autor segundo (ADOBE, 2006).

- **Gerenciamento de Fontes**: Consegue distinguir fontes para a utilização nos textos. Ao executar a conversão, o padrão PDF não trata necessariamente os textos como se fosse uma imagem, ele reconhece o texto e tenta formatá-lo o mais parecido possível com o arquivo de origem. Além disso, permite incorporar fontes e outros recursos ao arquivo PDF antes da sua geração;

- **Segurança**: Os arquivos PDF podem ser protegidos de diversas formas para restringir o acesso a seu conteúdo. As três formas principais de proteção de documentos são: criptografia de senha, criptografia de chave pública (certificado) e através do uso do servidor de políticas da Adobe;

- **Atualizações**: Permite que os usuários modifiquem um arquivo PDF original;

- **Extensibilidade**: O formato PDF foi projetado para ser extensível. Permite que novas características sejam adicionadas a um arquivo.

A partir destas características e por possuir suporte para praticamente todas as linguagens de desenvolvimento para Internet, o formato de arquivo PDF atende satisfatoriamente as principais necessidades dos desenvolvedores na geração de relatórios compactos, imprimíveis e atraentes.

1.1.3 Vantagens na utilização do PDF

O PDF é um formato universal de arquivo que preserva todas as fontes, formatação, cores e imagens gráficas de qualquer documento de origem, independente da aplicação e da plataforma utilizadas para criá-lo. Por sua estabilidade, confiabilidade e tamanho compacto, o PDF é hoje o formato mais moderno, prático e eficiente para envio de arquivos eletrônicos e uso gráfico (WELLING E THOMSON apud ADOBE, 2003).

Por se tratar de uma especificação de formato de arquivo aberto, o padrão PDF está disponível para qualquer pessoa que queira desenvolver ferramentas de criação, visualização ou manipulação de documentos PDF (ADOBE, 2007b).

Conforme Niederauer (2004) entre as principais vantagens desse formato, pode-se destacar o controle preciso sobre o posicionamento de elementos e, conseqüentemente, sobre

a impressão do documento. Possibilita também inserir imagens e referências no documento, seja através de *links*, marcadores ou miniaturas.

> [...] O formato PDF é ideal para a criação de qualquer documento que necessite de um controle preciso sobre sua impressão. Isso porque, ao contrário de outros formatos, em um documento PDF o que você vê na tela é o que você verá na impressora. Ou seja, a versão eletrônica e a impressa terão exatamente o mesmo aspecto [...] (NIEDERAUER, 2004, p.227).

Welling e Thomson (2003) citam outras vantagens do formato PDF: os documentos PDF oferecem uma saída consistente e de alta qualidade, são capazes de conter elementos como imagens de mapas de bit e imagens vetoriais, podem utilizar compactação para criar um arquivo pequeno, podem ser entregues eletronicamente e com baixo custo, são utilizáveis nos sistemas operacionais mais importantes e incluem controles de segurança. A tabela 1.2 apresenta, a seguir, os principais problemas enfrentados pelos formatos atuais de arquivos e soluções com a utilização do PDF.

Tabela 1.2 - Problemas enfrentados pelos formatos atuais de arquivos e soluções com a utilização do PDF.

Problemas comuns	**Soluções do PDF**
Os usuários não podem abrir os arquivos recebidos porque não dispõem dos aplicativos usados para criá-los.	O arquivo PDF pode ser aberto por qualquer pessoa, em qualquer lugar, necessitando apenas do *software* gratuito Adobe Reader.
A combinação de arquivamento em material impresso e eletrônico dificulta a pesquisa, toma espaço e requer que o aplicativo no qual um documento foi criado esteja disponível para acesso futuro.	Os arquivos PDF são compactos e de fácil pesquisa, podendo ser acessados usando o Adobe Reader. Os *hiperlinks* interativos facilitam a navegação pelos arquivos PDF.
Os documentos não são exibidos corretamente em dispositivos portáteis.	O PDF com marcas de formatação permite redistribuir o texto para exibição em plataformas móveis, tais como em dispositivos Palm OS, Symbian e Pocket PC.
As empresas dependem de material impresso para a troca de documentos e formulários por não disporem de processos eletrônicos que possam ser verificados e auditorados.	É possível atribuir direitos de acesso especiais e assinaturas digitais aos documentos PDF.
Documentos com formatação complexa não podem ser acessados por leitores com deficiências visuais.	Os arquivos PDF com marcas de formatação contêm informações sobre conteúdo e estrutura, o que permite que sejam acessados com a ajuda de leitores de tela.

Fonte: Adaptação do autor conforme (NIEDERAUER apud ADOBE, 2004).

Por essas e outras vantagens, o PDF tem se consagrado como o padrão de fato para a troca e distribuição de documentos e formulários eletrônicos no mundo todo, com segurança e confiabilidade.

1.1.4 Casos de Sucesso

O PDF tem sido adotado por empresas, educadores e governos do mundo todo para otimizar a troca de documentos, aumentar a produtividade e diminuir a dependência de papel (ADOBE, 2007b). A seguir, serão apresentados os casos sucesso da utilização do padrão PDF

nas empresas: Jornal O Estado de S. Paulo[5], Aché Laboratórios Farmacêuticos[6] e Seguradora Porto Seguro[7].

1.1.4.1 Jornal O Estado de S. Paulo

Segundo Gregório (2006b) desde o final de 2003 o jornal O Estado de S. Paulo, um dos jornais mais tradicionais do país, disponibiliza na Internet uma versão do jornal no formato PDF, com a mesma formatação das páginas impressas. As novas edições substituem as páginas anteriores, em HTML, que deixavam de fora partes importantes do conteúdo do jornal, como gráficos e imagens, além das páginas de cotações e os cadernos de classificados, entre outros.

A edição em PDF foi concebida para atender as necessidades do leitor que reside no exterior, aquele que precisa esperar muito tempo para receber a edição impressa. Com a opção do jornal no formato PDF, o leitor tem acesso imediato ao jornal do dia. Geralmente, essa versão é levada ao ar duas horas depois que a edição São Paulo do jornal é fechada, à meia-noite. Conforme Gregório (2006b), a empresa calcula que 40% dos assinantes do jornal, aproximadamente 120 mil pessoas, acessem a versão em PDF com maior ou menor freqüência.

A origem das páginas é a mesma do pacote que segue para a impressão na gráfica, vindas do sistema de editoração eletrônica do jornal. Para publicá-las na WEB, utiliza-se um sistema de gerenciamento de conteúdo desenvolvido em PHP, que roda em um servidor Linux. A edição em PDF é exibida por meio do *software* Adobe Reader, que funciona integrado ao *browser*. Possibilita a pesquisa de palavras no noticiário, fotos e ilustrações, cópia de textos e ajuste do nível de zoom para uma melhor visualização das páginas. O leitor pode também salvar o arquivo PDF com a página, guardando para uma futura visualização *offline*.

---

[5] JORNAL O ESTADO DE S. PAULO. Maiores informações em: http://www.estado.com.br.
[6] ACHÉ LABORATÓRIOS FARMACÊUTICOS. Maiores informações em: http://www.ache.com.br.
[7] SEGURADORA PORTO SEGURO. Maiores informações em: http://www.portoseguro.com.br.

Figura 1.2 - Edição do jornal O Estado de S. Paulo no formato PDF.
Fonte: http://www.estado.com.br.

1.1.4.2 Ache Laboratórios Farmacêuticos

Em uma indústria competitiva como a farmacêutica, pequenos ganhos de produtividade podem significar vantagens que fazem a diferença entre ganhar ou perder. Buscando melhorar o processo de aprovação de peças publicitárias, produção de dossiês e protocolos de pesquisa, a Aché Laboratórios Farmacêuticos encontrou nos documentos PDF um poderoso aliado.

De acordo com Alencar (2006) até o ano 2005 a aprovação das peças publicitárias da empresa demorava cerca de duas semanas. Cada uma dessas peças publicitárias passava pelas mãos de 15 revisores, entre gerentes e diretores. Os documentos eram impressos e agrupados em uma pasta comum. A deficiência também era grande na elaboração de documentos técnicos, como dossiês para a Anvisa[8] e protocolos contendo testes de novos medicamentos. Esses dossiês chegavam a ter mais de mil páginas, e são fundamentais para que os medicamentos fabricados recebam a aprovação do órgão governamental.

[8] ANVISA - Agência Nacional de Vigilância Sanitária. Maiores informações em: http://www.anvisa.gov.br.

Para acabar com esses problemas, a empresa implantou uma solução onde todos esses documentos são digitalizados pelos funcionários do Departamento de Documentação Científica da empresa, transformando-os em documentos PDF.

No caso das peças publicitárias, todos os revisores recebem, ao mesmo tempo, o arquivo no padrão PDF, via e-mail. Com o visualizador Adobe Reader, os profissionais envolvidos fazem suas anotações no texto, grifam, alteram e inserem comentários, e posteriormente enviam o documento com as anotações, também via e-mail, ao gerente responsável pelo processo. Esse método reduziu o prazo de aprovação para menos de cinco dias.

Já na elaboração dos dossiês e protocolos de pesquisa, o PDF permitiu um maior controle sobre as informações e a possibilidade de manter um arquivo na rede, e não em gavetas. Dessa forma, permite fácil localização de uma determinada informação há algum tempo arquivada.

1.1.4.3 Seguradora Porto Seguro

A Porto Seguro, uma das maiores seguradoras brasileiras, viu no formato PDF uma ferramenta simples para baixar custos e acelerar os processos de emissão de apólices de seguros. Há mais de 60 anos no mercado, a empresa está presente em todo o território nacional (com mais de 100 escritórios) e no Uruguai, trabalhando com mais de 18.000 corretores independentes cadastrados (PORTO SEGURO, 2007).

Conforme Gregório (2006a) antes de adotar o PDF a área de automóveis da Porto Seguro digitalizava as apólices em imagens no formato TIFF[9], um procedimento comum para arquivamento de documentos em corporações. A substituição deve-se à facilidade de manuseio, ao fato de o formato PDF ser um padrão universal e principalmente porque o tamanho dos arquivos TIFF são significativamente maiores que os equivalentes em PDF.

Para administrar o arquivo eletrônico de documentos, a equipe de TI da seguradora desenvolveu um aplicativo de gerenciamento capaz de fazer a leitura dos documentos PDF, chamado de Porto Viewer. Escritórios da Porto Seguro, espalhados por todo o país, possuem

[9] TIFF – Do inglês *Tagged Image File Format*. É um formato de arquivo digital para uso no processo de impressão PostScript. Transformou-se no formato padrão dos arquivos gráficos com elevada definição de cores.

acesso à mesma base de dados, por meio de um sistema de gerenciamento eletrônico de documentos (GED). Assim, fica mais fácil verificar informações sobre um determinado cliente, imprimir formulários e contratos ou visualizar o andamento dos negócios.

1.2 Outros formatos para geração de relatórios

Baseado em Welling e Thomson (2003), essa seção tem como objetivo apresentar os seguintes formatos de arquivo: ASCII, HTML, RTF e PostScript, também utilizados na geração de relatórios.

1.2.1 ASCII

ASCII acrônimo de *American Standard Code for Information Interchange*, representa uma maneira de codificar caracteres na forma de valores inteiros. Neste código, os caracteres são mapeados para valores numéricos representáveis por sete dígitos binários (*bits*). Este código abrange 95 caracteres passíveis de impressão e 33 caracteres especiais utilizados, entre outros, no controle de comunicação entre computadores ou entre um computador e seus periféricos.

Criar documentos no formato ASCII ou em texto simples pode proporcionar algumas vantagens: compatibilidade com os principais sistemas operacionais, baixa largura de banda necessária, criação e geração rápida de um *script*.

Ao disponibilizarmos um arquivo ASCII aos usuários teremos pouco controle sobre a sua aparência. Esse tipo de formato não possibilita controle de fontes, quebras de página, duplicação ou modificação de um documento, além de facilitar a alteração fraudulenta de qualquer arquivo.

1.2.2 HTML

HTML (*HyperText Markup Language*) é uma linguagem de marcação utilizada para produzir páginas na WEB, que inclui controle de formatação, sintaxe para incluir objetos como imagens e é compatível com uma variedade de sistemas operacionais e *softwares*.

Segundo Ruas (2002) o HTML não foi criado para controlar a aparência dos documentos, ao contrário dos processadores de texto e programas de *layout* de página. As

tags HTML apenas informam ao *browser* quais são os elementos que estão na página, elas dizem, por exemplo, que um determinado trecho é o título principal do documento, outro é um item de uma lista ou ainda uma célula de uma tabela.

As desvantagens da utilização do HTML na geração de relatórios são as seguintes: suporte limitado para impressão de formatos relacionados como quebras de página, pouca consistência da saída em plataformas ou navegadores diferentes e qualidade variável de impressão. Além disso, apesar de o HTML incluir qualquer tipo de elemento externo, a capacidade de o navegador exibir ou utilizar esses elementos não pode ser garantida para fontes incomuns.

1.2.3 RTF

Acrônimo de *Rich Text Format*, RTF é um formato de texto do Microsoft Word. Ele foi concebido como um formato para transferência de documentos entre diferentes programas. Semelhante ao HTML, utiliza sintaxe e palavras-chave em vez de dados binários para transportar informações de formatação.

Projetar um relatório no formato RTF apresenta algumas vantagens, das quais podemos citar: flexibilidade sobre o *layout* e formatação da página; possibilidade da utilização de uma variedade de elementos diferentes, como imagens vetoriais e mapa de *bits*; impressão de alta qualidade; facilidade e rapidez na geração.

No entanto, esse tipo de formato apresenta um problema extremamente crítico em um ambiente organizacional: a baixa confiabilidade. Um documento RTF pode ser livremente modificado por qualquer pessoa, fator capaz de inviabilizar a geração de relatórios corporativos nesse formato.

1.2.4 PostScript

O PostScript da Adobe Systems, é uma linguagem de descrição de página (*Page Description Language* – PDL) projetada para representar documentos de maneira independente de dispositivo, produzindo resultados consistentes entre dispositivos diferentes, como impressoras e monitores.

Um relatório no formato PostScript pode conter formatação precisa, texto, imagens, fontes embutidas e outros elementos. A descrição dos objetos contidos na página é gerada por um *driver* (tradutor) especial, e depois enviada ao dispositivo de impressão (um arquivo virtual ou uma impressora Postscript). Documentos PostScript são portáteis e fornecem impressões consistentes de alta qualidade a partir de diferentes dispositivos e sistemas operacionais.

Por outro lado, duas desvantagens significativas fazem com que esse tipo de formato não seja muito utilizado no processo de geração de relatórios corporativos:

- Os arquivos podem ser enormes;
- Muitos usuários precisarão fazer *download* de algum *software* adicional para visualizá-los, uma vez que esse tipo de arquivo não funciona no Windows sem a utilização de um visualizador específico.

1.3 Ferramentas para geração de relatórios

Dentre as inúmeras atribuições de um sistema de informação, a mais comum é a geração de relatórios. O processo de geração de um relatório pode ser resumido em duas fases, a definição da aparência (*layout*) do relatório e o mapeamento do banco de dados para os campos dentro do *layout* definido. Neste contexto, surgiram diversas ferramentas com intuito de auxiliar estas fases, tais como BIRT, Crystal Reports, JasperReports, Rave Reports, dentre outras. A seguir, o texto apresentará algumas destas ferramentas.

1.3.1 BIRT

BIRT (*Bussiness Intelligence and Reporting Tools Project*) é um projeto *open source* da Eclipse Foundation, que provê ferramentas para relatórios e *Business Intelligence* (BI)[10]. Foi projetado especialmente para aplicações WEB, principalmente as baseadas em J2EE (*Java 2 Enterprise Edition*), mas pode ser utilizado a partir de outras linguagens.

[10] *Business Intelligence* - É um conceito empregado a ferramentas, tecnologias e metodologias, que tem como objetivo fornecer informações estratégicas, que apóiam na tomada de decisão organizacional.

Com o BIRT é possível adicionar inúmeros relatórios a uma aplicação, dos quais podemos citar (BIRT, 2007):

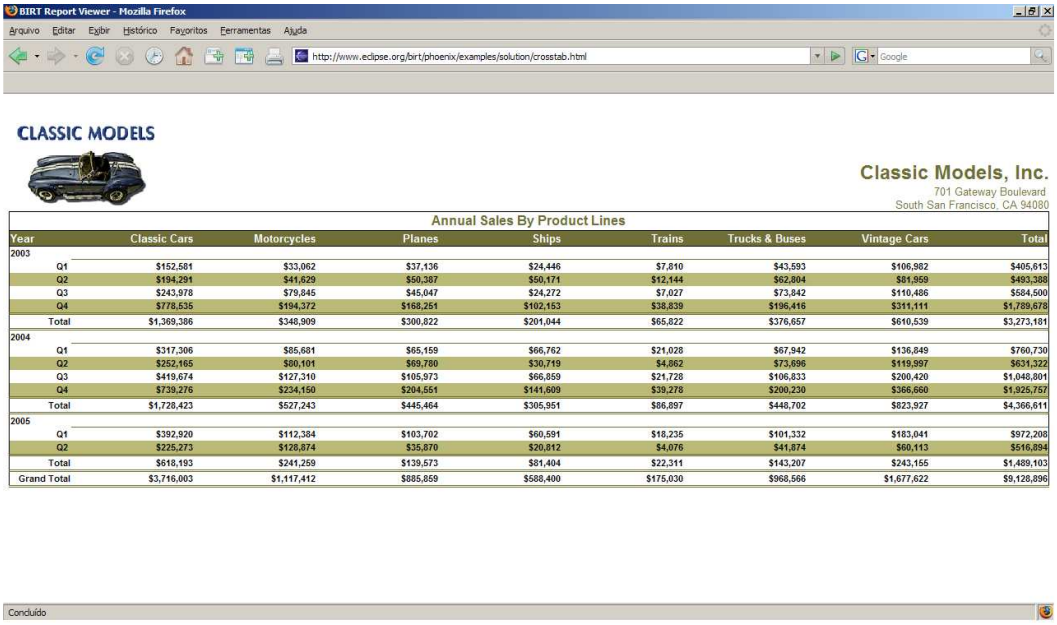
- **Lista de dados**: Permite adicionar grupos com o intuito de organizar listas de dados relacionados, como por exemplo, ordenar grupos por clientes ou agrupar produtos por fornecedor. No caso de dados numéricos, possibilita acrescentar totalizadores, médias e outros tipos de resumos;

- **Gráficos**: Dados numéricos são mais fáceis de serem entendidos se apresentados em forma de gráfico. O BIRT fornece gráficos de torta, linhas, barras, entre outros;

- **Matrizes**: Possibilita a exibição de dados em duas dimensões, como por exemplo, vendas por trimestre ou acessos em uma página WEB;

- **Cartas e documentos**: Notícias, formulários, cartas e outros documentos textuais podem ser criados no BIRT;

- **Relatórios compostos**: Muitos relatórios necessitam da combinação dos tipos citados acima em um único documento. Por exemplo, um relatório de clientes precisa listar as informações para cada cliente, prover texto sobre promoções atuais e apresentar uma lista lado a lado de pagamentos e taxas.



Annual Sales By Product Lines

Year	Classic Cars	Motorcycles	Planes	Ships	Trains	Trucks & Buses	Vintage Cars	Total
2003								
Q1	$152,581	$33,062	$37,136	$24,446	$7,810	$43,593	$106,982	$405,613
Q2	$194,291	$41,629	$50,387	$50,171	$12,144	$62,804	$81,959	$493,388
Q3	$243,978	$79,845	$45,047	$24,272	$7,027	$73,842	$110,486	$584,500
Q4	$778,535	$194,372	$168,251	$102,153	$38,839	$196,416	$311,111	$1,789,678
Total	$1,369,386	$348,909	$300,822	$201,044	$65,822	$376,657	$610,539	$3,273,181
2004								
Q1	$317,306	$85,681	$65,159	$66,762	$21,028	$67,942	$136,849	$760,730
Q2	$252,165	$80,101	$69,780	$30,719	$4,862	$73,696	$119,997	$631,322
Q3	$419,674	$127,310	$105,973	$66,859	$21,728	$106,833	$200,420	$1,048,801
Q4	$739,276	$234,150	$204,551	$141,609	$39,278	$200,230	$366,660	$1,925,757
Total	$1,728,423	$527,243	$445,464	$305,951	$86,897	$448,702	$823,927	$4,366,611
2005								
Q1	$392,920	$112,384	$103,702	$60,591	$18,235	$101,332	$183,041	$972,208
Q2	$225,273	$128,874	$35,870	$20,812	$4,076	$41,874	$60,113	$516,894
Total	$618,193	$241,259	$139,573	$81,404	$22,311	$143,207	$243,155	$1,489,103
Grand Total	$3,716,003	$1,117,412	$885,859	$588,400	$175,030	$968,566	$1,677,622	$9,128,896

Figura 1.3 - Exemplo de relatório WEB criado com o BIRT.
Fonte: Adaptação do autor segundo (BIRT, 2007).

1.3.2 Crystal Reports

O Crystal Reports é um aplicativo para geração de relatórios que pode acessar diversos tipos de base de dados ao mesmo tempo, desde simples bancos de dados locais até poderosos bancos em redes distribuídas (MONTOYA, 2005).

De acordo com Saade (2001), o Crystal Reports é considerada a melhor e mais poderosa ferramenta do mundo para criação, visualização e distribuição de relatórios. Pode ser utilizado de forma independente ou por meio de uma aplicação escrita em outra linguagem como Visual Basic, Delphi, Java, dentre outras.

Business Objects (2007) destaca como principais funcionalidades do Crystal Reports:

- Opções de agrupamento, inserção de fórmulas, totalizadores e sumarizações;
- Geração de relatórios a partir de uma interface WEB;
- Assistente para publicação de relatórios na WEB;
- Exportação de relatórios em diversos formatos, tais como, PDF, XLS, HMTL e RTF;
- Agendamento de relatórios.

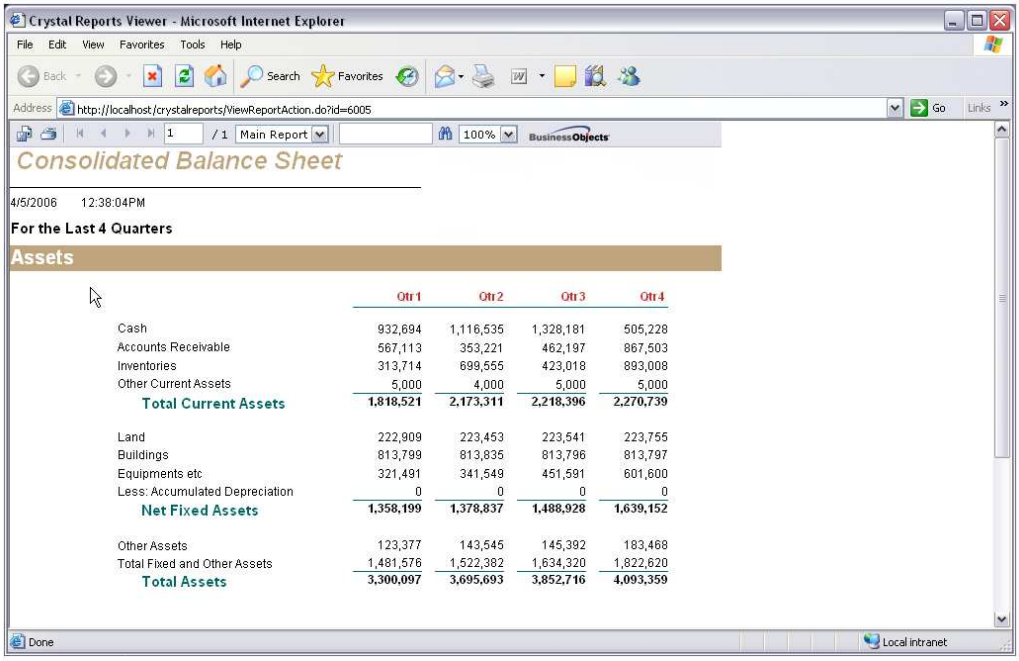


Figura 1.4 - Exemplo de um relatório WEB criado utilizando o Crystal Reports.
Fonte: Adaptação do autor conforme (BUSINESS OBJECTS, 2007).

1.3.3 JasperReports

JasperReports é um *framework open source* para geração de relatórios, escrito inteiramente em Java, que recebe como entrada uma descrição estruturada do relatório na forma de um documento XML[11], e gera a saída diretamente na impressora ou como um documento PDF, HTML, XLS ou CSV (LOZANO, 2005). Utilizando XML, o desenvolvedor pode definir textos estáticos, imagens, linhas, formas geométricas, e suas localizações dentro do relatório, ou ainda, definir os campos que serão preenchidos dinamicamente a partir de uma base de dados.

Facilmente integrado a aplicações Java empresariais, o JasperReports carece de um editor de relatórios visual integrado. Dessa forma, a geração de um relatório completo utilizando apenas o JasperReports pode representar uma tarefa longa, dolorosa e pouco compensadora. Felizmente, existem ferramentas visuais que geram descrições XML no formato esperado pelo JasperReports, executando-o em *background*, sem a necessidade de manipulação manual do código. Podemos citar como principais editores visuais o iReport[12] e o OpenReports[13].

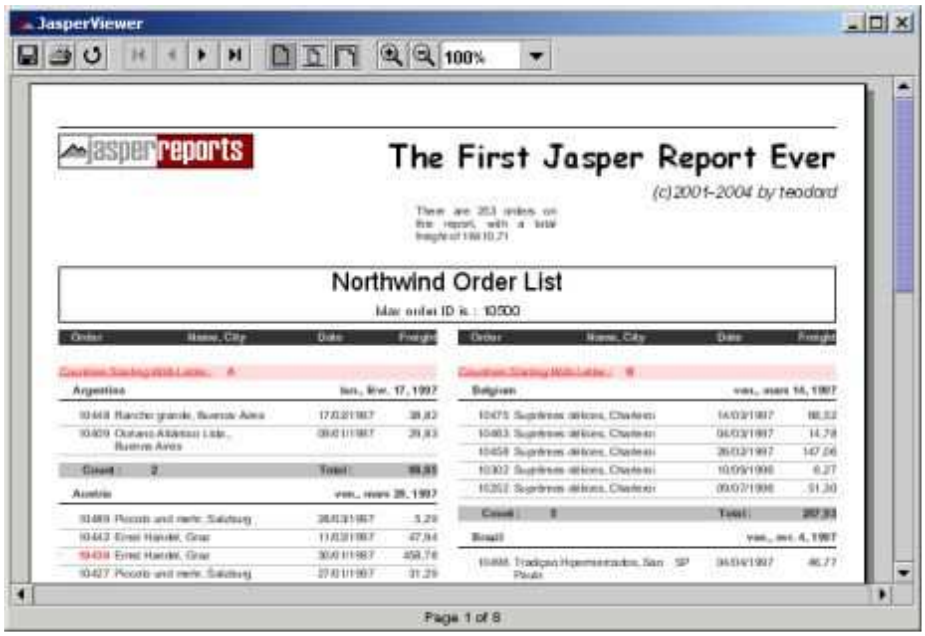


Figura 1.5 - Exemplo de um relatório criado utilizando o JasperReports.
Fonte: http:// jasperreports.sourceforge.net.

[11] XML – Do inglês *Extensible Markup Language*. É uma linguagem de marcação de dados que gera um formato para descrever dados estruturados, facilitando declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas (DAUM e UDO, 2002).
[12] IREPORT. Maiores informações em: http://jasperforge.org/sf/projects/ireport.
[13] OPENREPORTS. Maiores informações em: http://sourceforge.net/projects/oreports.

1.3.4 Rave Reports

Rave Reports é um gerenciador de relatórios desenvolvido e mantido pela empresa Nevrona Designs[14]. A partir do Delphi 7 o Rave Reports tornou-se a ferramenta padrão para geração de relatórios, substituindo o Quick Report[15].

De acordo com DevMedia (2007), apesar de ser distribuído junto ao Delphi, a partir da versão 7, o Rave Reports não é uma ferramenta nova no mercado, na verdade, é uma nova versão do Report Printer, ferramenta que está sendo mantida e desenvolvida desde a primeira versão do Delphi.

O RaveReports tem como principais características e funcionalidades:

- Possui um ambiente totalmente visual para criação de relatórios;
- Habilita o desenvolvimento de relatórios com acesso direto ao banco de dados, utilizando diversas tecnologias, como BDE, dbExpress, ADO e IBX. Possui ainda um editor visual para criação de instruções SQL integrado ao ambiente;
- Salva relatórios nos formatos RTF, HTML, PDF e texto. Possui ainda um formato proprietário (NDR);
- Possui diversas ferramentas para formatação dos relatórios, como alinhamento e posicionamento de objetos;
- Total acesso aos objetos do relatório a partir da aplicação Delphi;
- Possui um servidor WEB (versão *Server*) para disponibilizar relatórios em *browsers*;
- Disponibiliza um ambiente de desenvolvimento de relatórios para o usuário final, onde o usuário pode alterar e criar seus próprios relatórios.

[14] NEVRONA DESIGNS. Maiores informações em: http://www.nevrona.com.
[15] QUICK REPORT. Maiores informações em: http://www.qusoft.com.

1.4 Linguagens WEB que possuem recursos para a criação de relatórios PDF

Uma grande dificuldade encontrada pelos desenvolvedores WEB é proporcionar aos usuários a impressão de seus relatórios de forma correta e ordenada. Possibilitar a geração dos relatórios no formato PDF pode solucionar esse problema, pois esse tipo de arquivo oferece saída consistente e de alta qualidade, permitindo a geração de documentos imprimíveis e atraentes. A evolução das tecnologias WEB tem permitido que esse poderoso recurso esteja disponível em praticamente todas as linguagens de desenvolvimento para a Internet, dentre elas podemos citar PHP, ASP, Coldfusion e JSP.

1.4.1 PHP

O PHP (*Hypertext Preprocessor*) é uma linguagem *open source* que permite criar aplicações WEB dinâmicas, possibilitando uma interação com o usuário através de formulários, parâmetros de entrada, entre outras características. Por ser uma linguagem compatível com várias plataformas, o PHP possibilita a liberdade de escolha entre vários sistemas operacionais e servidores WEB, conforme mostra a tabela 1.3.

Tabela 1.3 - Sistemas operacionais e servidores WEB suportados pelo PHP.

Item	**Unix**	**Windows**
Sistema Operacional	AJX, A/UX, BSDI, Digital UNIX/Tru64, FreeBSD, HP-UX, IRIX, Linux, Mac OS X, NetBSD, OpenBSD, SCO UnixWare, Solaris, SunOS, Ultrix, Xerix.	Windows 95/98/ME, Windows NT/2000/XP/2003
Servidor WEB	Apache, fhttpd, Netscape	IIS, PWS, Netscape, Apache, Omni

Fonte: Adaptação do autor segundo (CONVERSE, PARKER e MORGAN, 2006).

Dall'Oglio (2003) destaca que o PHP é hoje a linguagem para soluções WEB mais utilizada no mundo. Altamente flexível, possui centenas de funcionalidades para os mais diferentes objetivos: manipulação de strings, arquivos e arrays, conectividade com uma série

de banco de dados (como Postgres, MySQL, Oracle, Sybase, Interbase, SQL Server, ODBC, Informix e Firebird), dentre outras.

A quantidade, diversidade e qualidade de seus recursos, assim como a facilidade de utilização foram os principais fatores que alavancaram o seu uso em grande escala. No início de 2003, o número de sites que utilizavam a linguagem PHP ultrapassava de dez milhões (NIEDERAUER apud NETCRAFT, 2004).

De acordo com Converse, Park e Morgan (2006), as principais vantagens da utilização do PHP são:

- **Custo**: é uma linguagem *open source*, portanto tem custo zero;
- **Licença de uso**: a liberdade de código-fonte aberto e *software* gratuito é garantido por um grupo de esquemas de licença, sendo o mais conhecido o GPL (*General Public License*);
- **Compatibilidade com sistemas operacionais**: o PHP pode ser utilizado na maioria dos sistemas operacionais, incluindo Linux, Microsoft Windows e MAC OS X;
- **Independência de servidor WEB**: suportado pela maioria dos servidores WEB, incluindo *Apache*, *Microsoft Internet Information Server* (IIS), *Microsoft Personal Web Server*, *Netscape Enterprise Server* e *AOLServer*;
- **Eficiência**: consome pouco recurso no servidor, tornando-se assim mais rápido;
- **Fácil aprendizado**: o PHP é fácil de aprender, pois tem uma sintaxe amigável e de fácil entendimento, além de possuir uma grande quantidade de fóruns de discussão, sites especializados, tutoriais, livros e demais mecanismos de aprendizagem.

Conforme Minetto (2007, p.14):

> Uma das grandes vantagens do PHP é sua facilidade de aprendizado. Ao ler poucas páginas de tutoriais ou de algum livro, um programador já é capaz de montar um formulário HTML e de criar um *script* PHP que processe os dados fornecidos pelo usuário. Isso favoreceu o rápido aumento do número de programadores e o surgimento de grandes *softwares*.

A tabela 1.4 a seguir, mostra um quadro com o comparativo de custos das linguagens WEB estudadas na seção.

Tabela 1.4 - Comparativo de custos das linguagens WEB estudadas na seção.

Item	**ASP**	**JSP**	**ColdFusion**	**PHP**
Desenvolvimento	US$ 0 – 480	US$ 0	US$ 395	US$ 0
Servidor	US$ 620	US$ 0 – 595	US$ 1.295	US$ 0
RDBMS	US$ 1.220 – 4.220	US$ 0 - ~ 10.000	US$ 0 - ~ 10.000	US$ 0
Suporte de Incidente	US$ 0 - 245	US$ 0 – 75	US$ 0 - 75	US$ 0

Fonte: Adaptação do autor segundo (CONVERSE, 2001).

Segundo Welling e Thomson (2003), o PHP disponibiliza suporte nativo para a criação de documentos PDF através de duas bibliotecas, de funções diferentes, mas com intenções semelhantes, a PDFlib e a ClibPDF. Por contarem com bibliotecas externas elas não são compiladas para o PHP por padrão.

Conforme Fuecks (2003), outra forma de geração de documentos PDF com PHP é a utilização das classes R&OS PDF Class e FPDF. Diferentemente da PDFlib e da ClibPDF, essas duas são classes escritas em PHP e não bibliotecas propriamente ditas.

Nas próximas seções, serão apresentados maiores detalhes das bibliotecas e classes citadas acima, mostrando suas vantagens e desvantagens, assim como sua forma de utilização.

1.4.2 ASP

ASP (*Active Server Pages*) é uma solução desenvolvida pela Microsoft que disponibiliza um conjunto de componentes para a criação de aplicações WEB dinâmicas e interativas. O desenvolvimento de páginas que usam ASP envolve a produção de um script HTML misturado com blocos de código de controle ASP. Este código de controle pode conter scripts em JavaScript ou VBScript (DEITEL, DEITEL e NIETO, 2003).

> Aderir ao ASP significa quase sempre optar pelo mundo Windows, uma vez que as páginas ASP só rodam nativamente em servidores Windows com servidor web IIS (*Internet Information Server*), da Microsoft. É possível rodar páginas ASP em sistemas Unix por intermédio de soluções comerciais, como o ChiliASP, ou usando o OpenASP, no servidor web Apache. Mas a implementação desse tipo de solução costuma ser problemática para os webmasters (CARDOSO, 2006, p.33).

A linguagem ASP permite a geração dinâmica de documentos no formato PDF através de componentes pagos, os quais devem ser instalados no servidor WEB. Dentre os principais componentes utilizados podemos citar: ASPPDF e ABCpdf.

ASPPDF é um componente ActiveX[16] de propriedade da Persits Software que permite ler, criar e modificar arquivos HTML para o formato PDF. Podemos citar como principais funcionalidades deste componente: leitura dos documentos em disco ou na base de dados; inserção de imagens nos formatos GIF, JPEG, BMP e TIFF; inclusão de texto formatado (posição, tamanho, tipo de letra, etc.); preenchimento de formulários; proteção de documentos com senha. Segundo ASPPDF (2007), a licença de uso do componente custa $299,00. No quadro 1.1 é apresentado um exemplo da criação de um arquivo PDF utilizando o componente ASPPDF.

Quadro 1.1 - Exemplo da geração de um arquivo PDF utilizando o componente ASPPDF.

```
<%
Set Pdf = Server.CreateObject("Persits.Pdf")
Set Doc = Pdf.CreateDocument
Doc.Title = " Demonstração do AspPDF "
Doc.Creator = " Igor Henrique Berlitz "
Set Page = Doc.Pages.Add
Set Font = Doc.Fonts("Helvetica")
Params = "x=0; y=650; width=612; alignment=center; size=50"
Page.Canvas.DrawText " Este é o componente AspPDF!", Params, Font
Filename = Doc.Save( Server.MapPath("asppdf.pdf"), False )
Response.Write " Download do arquivo PDF <A HREF=" & Filename & ">
aqui</A>"
%>
```

Fonte: Adaptação do autor segundo (ASPPDF, 2007).

O ABCpdf é uma solução proprietária da WebSupergoo Software que permite a conversão de arquivos escritos nas linguagens ASP, VBScript, VB.NET, C# e JavaScript para

[16] ActiveX - É uma conjunto de tecnologias (*software*) criado pela Microsoft para facilitar a integração entre diversas aplicações.

o formato PDF. Cabe ressaltar que esse componente é compatível com os sistemas operacionais Windows 2000, Windows NT, Windows XP e Windows 2003 e sua aquisição custa aproximadamente $329,00 (WEBSUPERGOO, 2007). A conversão de um arquivo ASP para formato PDF utilizando o componente ABCpdf pode ser feito conforme o exemplo mostrado no quadro 1.2.

Quadro 1.2 - Exemplo da criação de um arquivo PDF utilizando o componente ABCpdf.

```
<%
Set objPdf = Server.CreateObject("ABCpdf3.Doc")
objPdf.FontSize = 12
objPdf.Color = "255 0 0"
objPdf.Page = objPdf.AddPage()
theF1 = objPdf.AddFont("Helvetica")
objPdf.Font = theF1
objPdf.AddText "Este é o componente ABCpdf"
objPdf.Save "c:\temp\arquivo_abcpdf.pdf"
response.write " Download do arquivo PDF realizado com sucesso."
%>
```

Fonte: Adaptação do autor conforme (WEBSUPERGOO, 2007).

1.4.3 ColdFusion

ColdFusion é uma ferramenta de desenvolvimento para aplicações WEB que permite criar páginas dinâmicas através de uma integração entre elementos como banco de dados, ambiente WEB e aplicações de e-mail (OLIVEIRA, 2002).

O ColdFusion é composto de um Servidor de Aplicações ColdFusion (*Coldfusion Application Server*), uma linguagem (CFML - *ColdFusion Markup Language*) e de um ambiente de Administração.

O módulo *ColdFusion Application Server*, é uma solução proprietária da Adobe Systems que roda junto ao servidor WEB. Ele lê, interpreta e processa as instruções solicitadas. Essas instruções são passadas ao servidor através de páginas ColdFusion, escritas na linguagem CFML. Uma página ColdFusion é muito parecida com uma página gerada em HTML, porém contem *tags* específicas que somente o servidor ColdFusion pode ler, interpretar e retornar o resultado solicitado.

Atualmente na versão MX7, o ColdFusion possibilita a utilização de diversos recursos na geração de arquivos PDF, dentre os quais podemos destacar: definição do formato de página, criação de cabeçalhos e rodapés, inserção de quebras de páginas, proteção de arquivos por senha, entre outros. O trecho de código mostrado no quadro 1.3 exemplifica a geração de um arquivo PDF utilizando a linguagem ColdFusion.

Quadro 1.3 - Exemplo da geração de um arquivo PDF utilizando a linguagem ColdFusion.

```
<cfdocument
   format       = "PDF"
   filename     = "arquivo_cfml"
   unit         = "cm"
   marginbottom = "2"
   marginleft   = "3"
   marginright  = "2"
   margintop    = "3"
   orientation  = "portrait"
   userpassword = "password">
Criação de documento PDF com ColdFusion
</cfdocument>
```

Fonte: http://www.adobe.com/support/documentation/en/coldfusion/.

1.4.4 JSP

JSP, ou *JavaServer Pages*, é uma tecnologia utilizada no desenvolvimento de páginas WEB, a partir de componentes Java que executam no lado servidor. Essa tecnologia faz parte da plataforma J2EE e juntamente com os *JavaServlets* e *JavaBeans* pode ser usada no desenvolvimento de aplicações WEB.

A tecnologia JSP é uma extensão da tecnologia *JavaServlets*. Os *servlets* são classes Java, independentes de plataforma, que são encapsulados dentro da estrutura de um servidor WEB. Esta estrutura pode ser utilizada para ampliar as capacidades do servidor com os mínimos custos de processamento, manutenção e suporte. Diferentemente de outras linguagens de *script*, os *servlets* não se prendem a considerações ou modificações de uma plataforma específica; eles são componentes Java executados a pedido do sistema que necessite deles (SUN, 2007).

Para um melhor entendimento, podemos exemplificar da seguinte forma: quando uma página JSP é carregada pela primeira vez, o código Java é compilado gerando um *servlet*

que é executado, então cria uma página HTML que é enviada ao *browser*. As chamadas subseqüentes são enviadas diretamente ao *servlet* gerado na primeira requisição, sendo desnecessário uma nova etapa de geração e compilação do *servlet*.

A tecnologia JSP fornece um poderoso recurso na confecção de páginas WEB: as bibliotecas de *tags*, ou *taglibs*. Através delas, é possível encapsular trechos de código Java em tags, facilitando a manutenção, aumentando a clareza do código e estimulando o reuso das páginas (LEME, 2003).

A exportação de relatórios HTML para o formato PDF se torna uma tarefa menos árdua em JSP com a utilização das *taglibs*, uma vez que várias bibliotecas de *tags* disponíveis já possuem esse recurso implementado. Para um melhor entendimento, no quadro 1.4 será mostrado um exemplo de exportação de um relatório HTML para o formato PDF, utilizando a biblioteca *open source Display Tag Library*.

Quadro 1.4 - Exemplo da exportação de um relatório HTML para o formato PDF, utilizando a *Display Tag Library*.

```
<jsp:root version="1.2" xmlns:jsp="http://java.sun.com/JSP/Page"
xmlns:display="urn:jsptld:http://displaytag.sf.net/el"
xmlns:c="urn:jsptld:http://java.sun.com/jstl/core">
  <jsp:directive.page contentType="text/html; charset=UTF-8" />
  <jsp:directive.page import="org.displaytag.sample.*" />
  <jsp:include page="inc/header.jsp" flush="true" />
  <jsp:scriptlet>
    request.setAttribute( "test", new TestList(10, false) );
  </jsp:scriptlet>
  <h2>Exportação de Relatório</h2>
  <display:table name="test" export="true" id="currentRowObject">
     <display:column property="id" title="código" />
     <display:column property="email" />
     <display:column property="status" />
     <display:column property="date" title="data" />
     <display:setProperty name="export.pdf" value="true" />
  </display:table>
  <jsp:include page="inc/footer.jsp" flush="true" />
</jsp:root>
```

Fonte: http://displaytag.sourceforge.net.

1.5 Bibliotecas e classes PHP para a geração de relatórios PDF

A linguagem PHP, também objeto de estudo deste trabalho, nos possibilita diversos instrumentos para a criação e exibição de documentos PDF na WEB. Esse é um recurso bastante útil, que pode ser usado de acordo com o tipo de documento a ser entregue aos usuários. Uma das aplicações mais interessantes que pode ser criada com as funções PDF do PHP é a geração dinâmica de relatórios, baseada em informações obtidas de um banco de dados (NIEDERAUER, 2004).

Os relatórios fornecem várias formas de organizar, ordenar e resumir dados. Possuem características próprias, como divisão de seções (cabeçalho, rodapé, detalhes), campos calculados e apresentação de imagens. Freqüentemente são utilizados para apresentar uma descrição geral das informações relevantes, destacando fatos e tendências principais. Agrupar dados e ordená-los para que fiquem visivelmente mais explícitos são as vantagens chave dos relatórios, permitindo que as organizações tenham uma idéia geral das informações mais relevantes.

1.5.1 FPDF

FPDF é uma classe escrita em PHP que permite gerar documentos PDF diretamente da linguagem, sem a necessidade de usar a PDFlib, biblioteca nativa do PHP. Sua principal vantagem, diferentemente da PDFlib que necessita o pagamento de licença para uso comercial, é ser uma classe gratuita que pode ser usada para qualquer propósito e permite modificações conforme as necessidades do desenvolvedor (FPDF, 2007).

Diferentemente das bibliotecas nativas do PHP, como a PDFlib, que passa o objeto PDF como parâmetro para as funções, na FPDF as funções são chamadas como métodos em uma classe.

As principais características da classe, conforme FPDF (2007) são:

- Configuração da unidade de medida, formato da página e tamanho das margens;
- Personalização do cabeçalho e rodapé da página;
- Quebra de página automática;

- Quebra de linha e justificação de texto automático;
- Inserção de imagens nos formatos JPEG e PNG;
- Configuração de cores;
- Inserção de *links*;
- Suporte de fontes *TrueType* e *Type1*;
- Compressão de página.

1.5.1.1 Vantagens

Após descrever as características da classe FPDF, é importante apresentar as principais vantagens do seu uso na geração de arquivos PDF:

- Por se tratar de uma classe gratuita, permite utilização tanto para uso pessoal quanto comercial;
- Classe de simples manuseio, que possui inúmeras funcionalidades a serem exploradas, seja de maneira direta ou através de suas extensões;
- Possui métodos que podem ser sobrescritos, permitindo a herança de classes e implementação de métodos próprios;
- Possibilita o funcionamento em qualquer servidor PHP.

Segundo Niederauer (2004, p.228):

> A grande vantagem da FPDF é que ela está disponível na forma de uma classe PHP. Isso significa que você não precisa ter qualquer extensão instalada e configurada. Basta colocar os arquivos da classe em seu servidor e inserir uma chamada para ela no início dos seus programas.

1.5.1.2 Desvantagens

Mesmo trazendo diversos benefícios na geração de arquivos PDF, a classe FPDF apresenta alguns pontos negativos:

- Não possui uma interface de auxílio para o desenvolvimento automatizado de *layouts* de telas e relatórios a serem exportados;

- Necessita definir manualmente no código PHP a formatação e posicionamento do texto e de todos os elementos que formam o relatório;

- Não suporta texto na diagonal.

O objetivo desta seção foi apresentar algumas características, vantagens e desvantagens da classe FPDF. No capítulo 3 serão apresentados maiores detalhes da classe, através de exemplos práticos, mostrando as dificuldades enfrentadas pelos desenvolvedores WEB na geração de relatórios PDF fazendo uso da FPDF.

1.5.2 PDFlib

A PDFlib é uma biblioteca PHP criada por Thomas Merz, escrita em C, que fornece uma API de funções para gerar documentos PDF. Permite utilização não-comercial sem cobrança, mas exige o pagamento de uma taxa de licença caso seja utilizada em aplicações comerciais (PDFLIB, 2007).

A configuração da biblioteca no Unix/Linux consiste em compilar o PHP com suporte à PDFlib, enquanto que no Windows a extensão está disponível em um arquivo DLL, que precisa ser colocado no diretório das extensões PHP e habilitado no arquivo de configuração “php.ini”. As instruções detalhadas de configuração acompanham a extensão (NIEDERAUER, 2004).

Conforme Thomson e Welling (2003), a PDFlib não é estritamente parte do PHP, mas sim uma biblioteca separada, com várias funções projetadas para serem chamadas a partir de uma ampla variedade de linguagens de programação. As vinculações de linguagem estão disponíveis para C, C++, Java, Perl, Python, Tcl e ActiveX/Com.

Todas as funções da PDFlib e do módulo PHP possuem nomes iguais para funções e parâmetros. Thomson e Welling (2003) destaca ainda, que a PDFlib trabalha com pontos, tanto para tamanho de página como para posicionar as coordenadas em cada página. Para utilizar imagens TIFF ou JPEG é necessária a instalação das suas respectivas bibliotecas.

A PDFlib possui suporte para 14 tipos de fontes diferentes. Fontes fora desse conjunto podem ser inseridas nos documentos, no entanto, além de aumentar consideravelmente o tamanho do arquivo, a licença do produto pode não permitir que uma

determinada fonte seja utilizada (LERDOF e TRATOE, 2002). A tabela 1.5 lista as fontes suportadas pela biblioteca.

Tabela 1.5 - Fontes suportadas pela PDFlib.

Fontes suportadas			
Courier	Courier-Bold	Courier-BoldOblique	Courier-Oblique
Helvetica	Helvetica-Bold	Helvetica-BoldOblique	Helvetica-Oblique
Times-Bold Symbol	Times BoldItalic	Times-Italic	Times-Roman
Symbol	ZapfDingbats		

Fonte: Adaptação do autor segundo (LERDOF e TRETOE, 2002).

1.5.3 ClibPDF

ClibPDF é uma biblioteca alternativa criada pela empresa FastIO e está disponível em forma de extensão igual a PDFlib. Segundo PHP (2007), suas funcionalidades e interface de programação são semelhantes à PDFlib, sendo que muitas de suas funções possuem o mesmo nome.

PHP (2007) destaca como principais características da ClibPDF:

- Não suporta a criação de vários documentos PDF em uma única vez;
- Possibilita a de criação de um documento PDF completo alocado em memória, sem a necessidade de usar arquivos temporários;
- Qualquer página pode ser modificada a qualquer momento, inclusive se uma nova página já tiver sido aberta.

Para utilizar as funções da ClibPDF é necessário instalar o pacote ClibPDF. Esse pacote é de propriedade da empresa Fastio e necessita a aquisição de uma licença para o uso comercial.

1.5.4 R&OS Class

R&OS Class é uma biblioteca *open source* desenvolvida por Wayne Munro, que atualmente encontra-se na versão 0.9. Possui funções que possibilitam a utilização de formas geométricas, imagens e *links* nos documentos PDF a serem gerados. Por não fazer parte da

documentação original do PHP, essa classe não é muito utilizada pelos desenvolvedores na geração de documentos PDF, apesar de simples e bem documentada (FUECKS, 2003).

2 A NOVA GERAÇÃO DAS APLICAÇÕES WEB

Em menos de 15 anos, a Internet – e com ela o ambiente em rede a longa distância – surgiu como uma nova ferramenta de comunicação acessível a usuários domésticos e empresariais. Saltando do uso restrito a alguns poucos pesquisadores em centros de pesquisa, passou a crescer a uma taxa média anual de 14%, até atingir cerca de 1 bilhão de usuários no final de 2005 (CAVALCANTI e NEPOMUCENO apud EMARKETER, 2007). O crescimento explosivo da Internet é um fenômeno revolucionário em computação e telecomunicações. Definitivamente, pode-se afirmar que a Internet abriu as portas para o desenvolvimento da sociedade da informação e do conhecimento.

2.1 WEB 2.0

O aproveitamento das oportunidades abertas pela Internet continua em ritmo acelerado, com o surgimento das redes sociais, como MySpace e Orkut, de *softwares* de pesquisa sofisticados, o principal deles o Google, sites de comparação de produtos e preços, aplicações de mensagens instantâneas, WEB *chats*, Skype, que em conjunto com *softwares* de protocolos abertos como Linux, Apache, MySQL e PHP, deram origem a um novo ambiente de TI que tem sido chamado de WEB 2.0, na qual as comunidades em rede têm um papel central (FONSECA apud CAVALCANTI e NEPOMUCENO, 2007).

2.1.1 O que é WEB 2.0

A expressão WEB 2.0 surgiu em outubro de 2004, durante uma sessão de *brainstorm*[17] entre representantes das empresas O'Reilly Media[18] e MediaLive[19], com a finalidade de reunir, integrar e compreender uma série de fenômenos e ações que, vistos em conjunto, formavam um novo cenário, uma nova fase, uma nova versão da Internet (CAVALCANTI e NEPOMUCENO, 2007).

Conforme O'Reilly (2005) WEB 2.0 é a mudança para uma Internet como plataforma, e um entendimento das regras para obter sucesso nessa nova plataforma. Entre outras, a regra mais importante é desenvolver aplicativos que aproveitem os efeitos de rede para se tornarem melhores quanto mais são usados pelas pessoas, aproveitando a inteligência coletiva.

Musser (2006) define WEB 2.0 como um conjunto de tendências econômicas, sociais e tecnológicas que servem como base para uma nova geração da Internet – mais madura, caracterizada pela participação do usuário, efeitos de rede e inteligência coletiva.

Para Cavacalti e Nepomuceno (2007) a WEB 2.0 é um conceito para agrupar, nomear e incentivar projetos que expandem o principal potencial do ambiente de rede – um novo meio voltado para a interação e capaz de implementar novas formas de produzir conhecimento: a Inteligência Coletiva em rede.

De acordo com Carter (2007) o termo WEB 2.0 refere-se à segunda geração de serviços na Internet com ênfase na colaboração e troca de informações *online*. As aplicações desta geração disponibilizam interfaces dinâmicas, muito parecidas com as aplicações *desktop*, em contraposição com as páginas praticamente estáticas da primeira geração de aplicações para WEB.

---

[17] *Brainstorm* - Dinâmica de grupo em que as pessoas, de forma organizada e cooperativa, com oportunidades iguais, podem dar opiniões sobre determinado problema, encontrando as causas e sugerindo soluções.
[18] O'REILLY MEDIA. Maiores informações: http://www.oreilly.com.
[19] MEDIALIVE. Maiores informações em: http://www.cmp.com.

2.1.2 A WEB 2.0 e os conceitos envolvidos

Desde sua primeira aparição, o termo WEB 2.0 tem causado polêmica e divisões entre grupos. Muitos especialistas consideram que não existe uma segunda geração de aplicativos WEB, e sim uma evolução natural destes, acreditando que essa nomenclatura não passa de uma estratégia de marketing da O'Reilly Media, uma vez que não ocorreram mudanças tecnológicas significativas na grande rede.

Nesse contexto, a conceitualização dada nesta seção segue os sete princípios fundamentais da WEB 2.0 ditados por Tim O'Reilly, conhecido como o precursor do uso do termo, no artigo intitulado "*What is Web 2.0*" (O'REILLY, 2005):

- **A WEB como plataforma**: O conceito de plataforma WEB se baseia na idéia de aplicações que são desenvolvidas utilizando serviços disponíveis na Internet. Assim a WEB funciona de forma semelhante a um sistema operacional, como o Windows, que fornece serviços para que terceiros possam desenvolver novas aplicações e funcionalidades.

- **Empregando a inteligência coletiva**: Inteligência coletiva é uma nova forma de produzir conhecimento em rede, através de conexões sociais e ações dirigidas por comunidades que se utilizam ou se apropriam de ferramentas interativas disponíveis nos ambientes de rede (CAVALCANTI e NEPOMUCENO, 2007). Sites colaborativos e redes sociais, onde o usuário produz e organiza todo o conteúdo - como o Del.icio.us, Flickr, Orkut e Digg - são bons exemplos de como gerar inteligência coletiva em rede.

- **Dados são o próximo Intel Inside**: Neste novo cenário, as aplicações são totalmente voltadas à captação e armazenagem de dados. Portanto, para obter uma vantagem competitiva é necessário possuir uma base de dados única e difícil de ser recriada.

- **O fim do ciclo de lançamento de *softwares***: Na WEB 2.0 acabam os ciclos de lançamento de *softwares*. Nessa nova era da Internet, o *software* passa a ser apresentado não mais como um produto, e sim como um serviço. Conseqüentemente, os programas são alterados, corrigidos e melhorados de maneira constante, com a

participação direta dos usuários neste processo, dando sugestões, reportando erros e aproveitando as melhorias.

- **Modelos leves de programação**: Ao invés de desenvolver um grande e complexo *software*, na WEB 2.0 são implementados *softwares* simples e modulares. Em tempos de mudanças rápidas, ter uma estrutura de *software* enxuta, que permita ser expandida em módulos e aperfeiçoada ao mesmo tempo em que está sendo usada, torna-se um diferencial positivo.

- ***Software* em mais de um dispositivo**: Outra grande vantagem da utilização da WEB 2.0, é que os usuários não dependerão mais de um dispositivo físico específico para acessar aos aplicativos WEB. As aplicações são projetadas para serem acessadas por dispositivos móveis, PC's, servidores WEB, entre outros.

- **Experiência rica do usuário**: A WEB 2.0 propõe uma experiência de uso semelhante à de aplicativos para *desktop*, freqüentemente fazendo uso de uma combinação de tecnologias, que incluem *WEB Services*, AJAX, RSS, entre outras. Estas tecnologias aumentaram a velocidade e a facilidade de uso de aplicativos WEB, sendo responsáveis por um aumento significativo no conteúdo, seja ele colaborativo ou meramente expositivo, existente na Internet.

A figura 2.1 a seguir, apresenta um mapa de noções WEB 2.0, onde O'Reilly (2005) define que a WEB 2.0 não tem limites rígidos, mas sim um núcleo gravitacional, sendo encarada como uma plataforma, na qual os próprios usuários controlam seus dados. O que caracteriza essa nova WEB é a existência de serviços e não de pacotes fechados de *software* e sua arquitetura é edificada sob a cooperação.

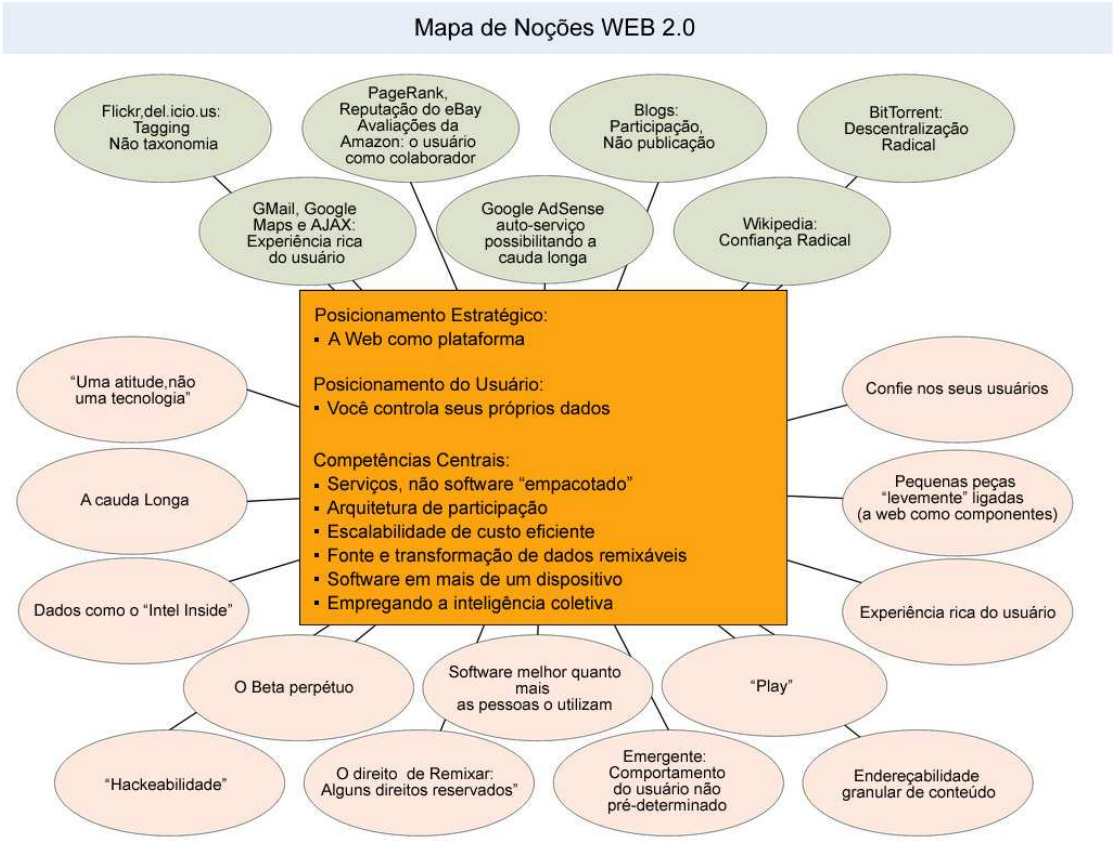


Figura 2.1 - WEB 2.0 e os conceitos envolvidos.
Fonte: Adaptação do autor segundo (O'REILLY, 2005).

2.1.3 Tecnologias da WEB 2.0

A infra-estrutura da WEB 2.0 é complexa e está em evolução. Ela inclui *softwares* baseados em servidores, protocolos de comunicação, *browsers* baseados em padrões e várias aplicações cliente. Estas abordagens diferentes e complementares provêem a WEB 2.0 com capacidades de criação, disseminação e armazenamento de informações além da imaginada inicialmente para páginas WEB. Baseado em Musser (2006) serão apresentadas algumas tecnologias comumente utilizadas em sites WEB 2.0, das quais podemos destacar, AJAX, RSS, SOA e *WEB Services*.

AJAX, acrônimo de *Asyncronous Javascript and XML*, é o uso sistemático de Javascript e XML para tornar o *browser* mais interativo com o usuário. O AJAX não é uma tecnologia, são na verdade várias tecnologias progredindo de forma independente, e que se juntaram a fim de explorar formas de melhorar a interação com os usuários em aplicações WEB (CRANE, PASCARELLO e JAMES, 2006). Na próxima seção serão apresentados maiores detalhes da tecnologia AJAX, mostrando as vantagens da utilização da mesma no desenvolvimento de aplicações para Internet.

O **RSS** (*Really Simple Syndication*) é uma especificação para a distribuição de conteúdo através da utilização da linguagem XML. A especificação surgiu com o objetivo de padronizar a forma de distribuição de conteúdos estruturados existentes nos sites WEB. O usuário tem acesso a esse conteúdo através dos leitores ou agregadores de RSS, que podem ser *online*, utilizando a WEB como plataforma, podem ser *softwares* instalados no computador, ou ainda já vir embutidos nos navegadores (HAMMERSLEY, 2003). A figura 2.2 a seguir, mostra o exemplo da utilização da tecnologia RSS no portal de notícias da Globo.com.

Figura 2.2 - Exemplo da utilização da tecnologia RSS no portal de notícias da Globo.com.
Fonte: http://g1.globo.com/

SOA (*Service Oriented Architecture*), segundo Carter (2007), é um modelo de componente na qual as aplicações fazem uso de serviços disponíveis em uma rede, como a WEB. Implementar uma arquitetura SOA envolve o desenvolvimento de aplicações que usam serviços ou que são disponibilizados como serviços, de forma que outras aplicações possam usar esses serviços ou vice-versa. Assim, pode-se dizer que SOA é uma forma de compartilhar serviços de maneira abrangente e flexível, e esta é exatamente a sua principal característica para a WEB 2.0.

WEB Services são aplicações implementadas estaticamente ou dinamicamente, através de tecnologias da Internet, que formam um conjunto de especificações de interface independentes do mecanismo de transporte, da arquitetura de *hardware* ou sistema

operacional, baseado nos cinco padrões definidos pela W3C[20]: XML, WSDL[21], SOAP[22], UDDI[23] e HTTP[24] (BASSIURA et al., 2003).

2.1.4 Ferramentas WEB 2.0

Segundo Cipriani (2006), atualmente vivemos uma nova era. Uma nova era com a consolidação da Internet como o grande repositório de dados e agente transformador de processos e meios de comunicação. Uma nova era dentro da própria Internet, que por ser global e encurtadora de distâncias, viabilizou a contribuição, opinião e inteligência em massa.

Baseado em autores como (CAVALCANTI e NEPOMUCENO, 2007; CIPRIANI 2006), a seguir serão apresentados alguns exemplos de ferramentas e projetos interativos que utilizam conceitos relacionados à WEB 2.0:

Flickr é um serviço pertencente a Yahoo que possibilita o compartilhamento, busca, armazenamento e classificação de arquivos de imagem, com palavras-chave, na Internet. O serviço também possibilita a formação de comunidades que agrupam usuários com interesses comuns. Flickr possui uma API que permite que aplicações de terceiros utilizem sua base de dados (FLICKR, 2007).

Para compartilhar *links* favoritos e indicar os melhores sites para assuntos classificados por palavras-chave o **Delicious** é a ferramenta mais popular. Precursor desse tipo de serviço, o Delicious tem uma interface bastante simples, rápida e direta. Pode ser acessado através do endereço: http://del.icio.us.

De acordo com Cipriani (2006), a WEB 2.0 nos oferece algumas ferramentas para colocarmos em prática aquilo que o ser humano vem praticando há milênios: a vida em comunidade. O maior exemplo desse desejo de se comunicar explica a razão de ser do **Orkut**

---

[20] W3C - Consórcio internacional que regulamenta e define padrões para a WEB. Maiores informações em: http://www.w3.org.
[21] WSDL - Do inglês *Web Services Description Language*. É uma linguagem baseada em XML que descreve os dados e serviços oferecidos por um *WEB Service*.
[22] SOAP - Do inglês *Simple Object Access Protocol*. É o protocolo de transporte utilizado na troca de dados entre o provedor e o consumidor de um *WEB Service*.
[23] UDDI - Do inglês *Universal Description and Discovery of Information*. É um registro no qual WEB Services podem ser publicados e pesquisados
[24] HTTP - Do inglês *HyperText Transfer Protocol* (Protocolo de Transferência de Hipertexto). É um protocolo da camada de aplicação do modelo OSI, utilizado para transferência de dados na WEB.

(http://www.orkut.com), a página de criação de comunidades e contatos, que cresceu absurdamente no Brasil. Isso não significa, no entanto, que só brasileiros gostam de fazer comunidades. Na verdade, cada país acaba se encontrando no ambiente que proporciona um maior número de *links*. Outras ferramentas que se destacam são o MySpace[25], o Friendster[26] e o Facebook[27].

Conforme Cipriani (2006) poucos fenômenos resumem tão bem o espírito da WEB 2.0 e seu efeito multiplicador quanto ao site de compartilhamento de vídeos **YouTube**. Consiste basicamente em um compartilhador de vídeos pessoais, onde o usuário pode fazer o *upload* de seu vídeo e permitir que pessoas do mundo inteiro assistam, comentem, dêem nota, dentre outras possibilidades, como pegar o código HTML de qualquer vídeo para exibi-lo em sua própria página WEB. O endereço para acessar o Youtube é: http://www.youtube.com.

O **ThinkFree** utiliza o conceito de *software* online. Trata-se de um grupo de ferramentas de escritório que utilizam a WEB como plataforma. Todos os aplicativos contam com uma interface próxima de programas para *desktop*. Na melhor linha da WEB 2.0, o ThinkFree conta com integração com outros serviços, podendo publicar textos em *blogs* e usar fotos provenientes do Flickr em qualquer de seus aplicativos. O ThinkFree pode ser acesso através do endereço: http:// www.thinkfree.com.

2.2 AJAX

A WEB 2.0 propõe uma experiência de uso semelhante à de aplicativos para *desktop*, freqüentemente fazendo uso de uma combinação de tecnologias surgidas no final da década de 1990, que incluem *WEB Services*, API, AJAX, RSS, entre outras. Estas tecnologias aumentaram a velocidade e a facilidade de uso de aplicativos WEB, sendo responsáveis por um aumento significativo no conteúdo existente na Internet (O'REILLY, 2005).

O AJAX surgiu como um protagonista da WEB 2.0, pois ele modifica o modo como os *browsers* interagem com as informações disponíveis na Internet. Portanto, pode-se dizer que o AJAX é um dos primeiros passos dessa nova geração da Internet. Conhecendo o conceito de WEB 2.0, fica clara a importância do AJAX nesse processo, pois ele diminui

[25] MYSPACE - Maiores informações em: http://www.myspace.com.
[26] FRIENDSTER - Maiores informações em: http:// www.friendster.com.
[27] FACEBOOK - Maiores informações em: www.facebook.com.

muito a distância entre aplicações para *desktop* e as aplicações para a WEB (NIEDERAUER, 2007).

2.2.1 O que é AJAX?

AJAX, acrônimo de *Asyncronous Javascript and XML*, é o uso sistemático de Javascript e XML, entre outras tecnologias, com o intuito de tornar o *browser* mais interativo com o usuário, utilizando-se de solicitações assíncronas de informações. AJAX não é somente um novo modelo, é também uma iniciativa na construção de aplicações WEB mais dinâmicas e criativas (NIEDERAUER, 2007).

O termo AJAX foi cunhado por Jesse James Garrett, em fevereiro de 2005, em seu artigo intitulado "*AJAX: A New Approach to Web Applications*" (GARRET, 2005), e ganhou popularidade ao mostrar-se um bom termo para descrever técnicas que permitem aplicações WEB interagir com um servidor assincronamente (ZAKAS, MCPEAK e FAWCETT, 2006).

De acordo com Niederauer (2007), desde o surgimento da Internet, o modelo de interação entre usuário e servidor via HTTP é baseado em um sistema simples de hipertexto, ou seja, o usuário clica em um *link* para requisitar um documento, o servidor responde, processando sua requisição e devolvendo-lhe uma página HTML por completo. A cada solicitação tem-se a necessidade de executar novamente todo o processo de envio, criação de página e devolução do conteúdo ao *browser*. Estas atividades geram um tráfego excessivo de rede e envio desnecessário de dados que não foram alterados. A conseqüência desse processo para o usuário, muitas vezes, é uma tela lenta e páginas em branco.

O modelo AJAX adiciona uma camada intermediária entre o cliente e o servidor. A interface, ao invés de enviar uma solicitação HTTP, lança uma chamada de Javascript para a aplicação AJAX. O servidor processa a solicitação e envia uma resposta. Caso o servidor retorne dados, o AJAX poderá utilizar esses dados para fazer a atualização apenas de uma parte da página que o usuário está visualizando, sem a necessidade de recarregá-la totalmente. Caso contrário, o usuário também poderá continuar usufruindo normalmente da página, mas ela não sofrerá qualquer alteração visual (GARRET, 2005).

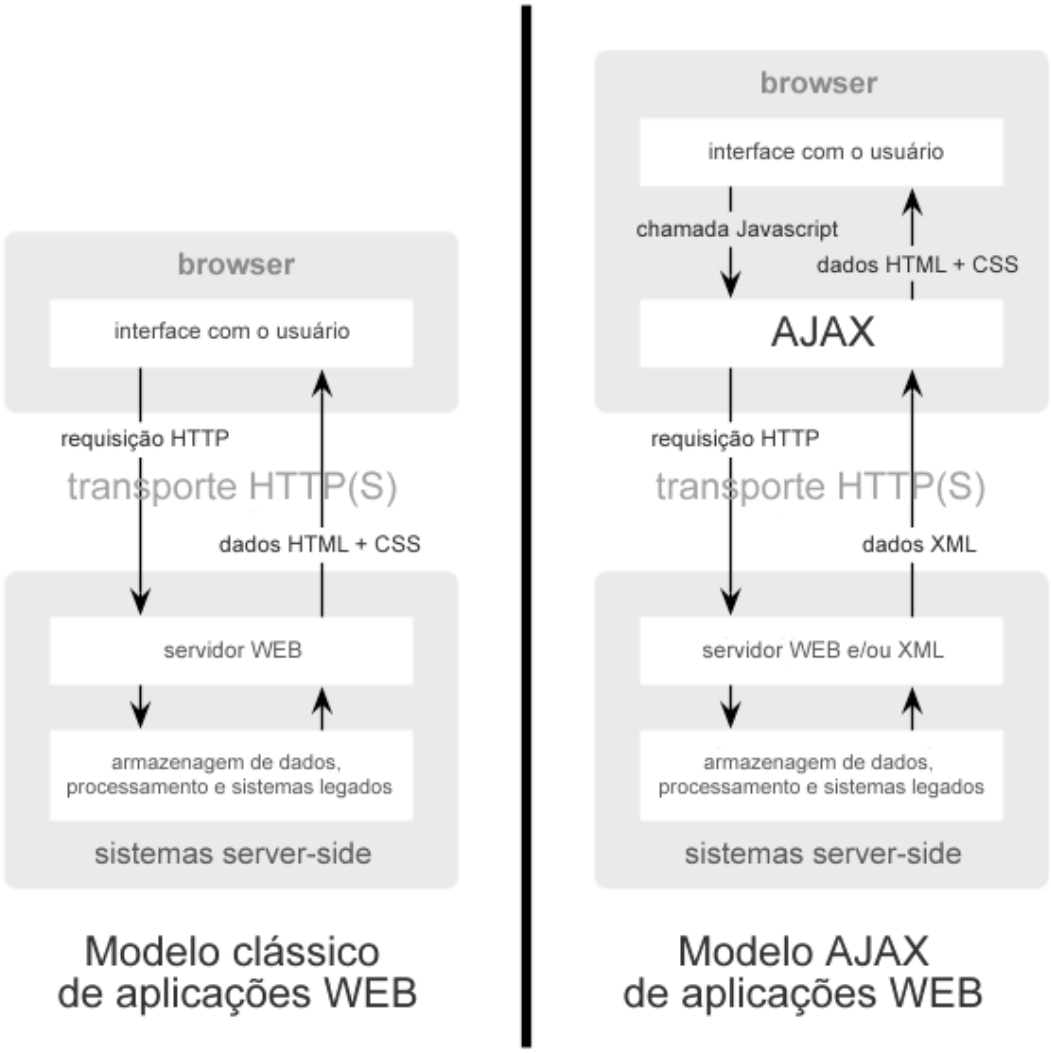


Figura 2.3 - Comparativo entre o modelo clássico e o modelo AJAX.
Fonte: Adaptação do autor segundo (Garret, 2005).

Na figura 2.3, percebe-se bem a diferença entre os dois modelos e os módulos envolvidos. Á esquerda, no modelo clássico de navegação têm-se a necessidade de recarregar a página HTML por completo, cada vez que é feito um pedido ao servidor. Já no modelo AJAX, mostrado a direita, uma vez que o pedido foi solicitado no carregamento inicial, serão alterados apenas os dados na área onde os mesmos serão apresentados, possibilitando que o usuário continue trabalhando enquanto aguarda o retorno.

2.2.2 Tecnologias envolvidas

Segundo Garret (2005), o AJAX não é uma tecnologia, são na realidade várias tecnologias progredindo de forma independente, e que se juntaram a fim de explorar formas de melhorar a interação com os usuários em aplicações WEB. AJAX incorpora em seu modelo:

- Padrões de apresentação baseada em *Web Standards*[28], utilizando XHTML e CSS;
- Apresentações dinâmicas e interativas usando o *Document Object Model* (DOM);
- Manipulação de elementos de uma página utilizando XML e XSLT;
- Recuperação assíncrona de dados usando o objeto XMLHttpRequest;
- e Javascript, fazendo a integração das anteriores.

XHTML (*Extensible Hypertext Markup Language*) é uma coleção de tipos e módulos de documento especificados pela W3C, que reproduz, subdivide e estende a linguagem HTML. Os tipos de documentos são baseados na XML, ou seja, XHTML é uma versão da HTML que usa sintaxe XML (DAUM e UDO, 2002).

Conforme DAUM e UDO (2002), a idéia básica por trás da XHTML é oferecer aos desenvolvedores WEB um padrão que permita a criação de páginas WEB mais ricas em uma ampla variedade de dispositivos e plataformas.

CSS (*Cascading Style Sheets*), ou folhas de estilos em cascata, é uma linguagem de estilo utilizada para definir a apresentação de documentos escritos em linguagem de marcação, como HTML ou XML. Seu principal benefício é prover a separação entre o formato e conteúdo de um documento (MAHEMOFF, 2006).

Utilizando CSS é possível definir padrões de aparência e comportamento dos elementos de uma página. Cores, parágrafos, margens, espaçamentos, espessura ou qualquer outro elemento podem ser definidos em apenas um arquivo, possibilitando alterações mais rápidas e fáceis no visual do site.

DOM ou *Document Object Model* é uma interface de programação de aplicativos (API) completa para documentos XML. Essa interface permite que os clientes não apenas naveguem dentro de documentos XML, mas também criem, modifiquem ou excluam elementos e conteúdo (DAUM e UDO, 2002).

---

[28] *Web Standards* – Conjunto de normas definidas pelo W3C e destinadas a orientar a criação de WEB Sites acessíveis a todos, independentemente dos dispositivos usados ou de suas necessidades especiais.

A API DOM oferece uma maneira padrão de acessar os elementos de um documento, além de permitir trabalhar com cada um desses elementos separadamente, e por esses motivos criar páginas altamente dinâmicas. De acordo com Niederauer (2007), o modelo DOM entra no processo do AJAX na parte de interação dinâmica, sendo utilizado no tratamento dos dados retornados pelo servidor, ou seja, é a tecnologia que irá conferir dinamismo, aperfeiçoando a manipulação das informações em questão.

XML (*Extensible Markup Language*) é uma linguagem de marcação de dados que gera um formato para descrever dados estruturados, facilitando declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas (DAUM e UDO, 2002).

Observando o diagrama das tecnologias envolvidas no AJAX, apresentado na figura 2.4 a seguir, percebe-se que o XML tem como função a troca e manipulação dos dados, ou seja, é a linguagem que o servidor WEB utiliza para retornar os dados solicitados pela aplicação. No entanto, esse retorno não precisa ser necessariamente em formato XML, podendo ser inclusive um fluxo de texto simples.

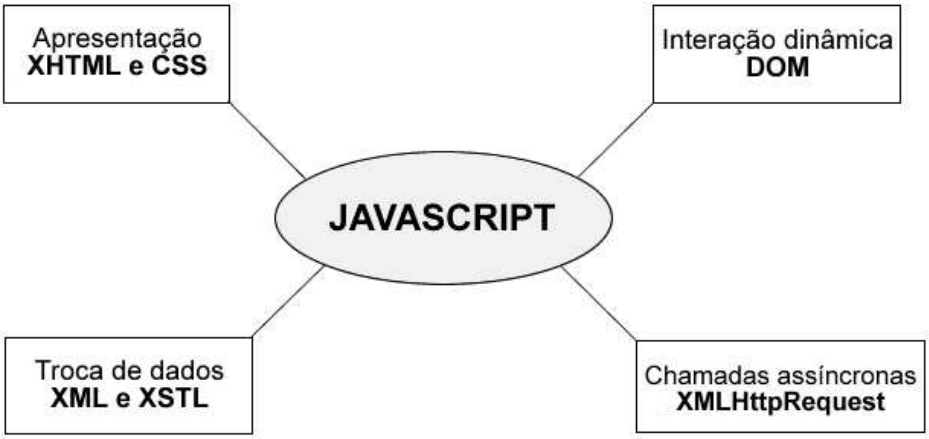


Figura 2.4 - Tecnologias utilizadas pelo AJAX.
Fonte: Adaptação do autor segundo (NIEDERAUER, 2007).

Segundo Niederauer (2007), a preferência pela utilização dessa linguagem, deve-se principalmente à facilidade na manipulação dos dados de um arquivo XML, pois esse formato disponibiliza dados estruturados e, conseqüentemente, uma aplicação mais organizada e um código-fonte mais fácil de ser mantido e acessado.

XSLT ou *Extensible Stylesheet Language Transformation* é uma linguagem versátil utilizada para transformar documentos XML em outros formatos, como por exemplo, HTML

para exibição na WEB. Com o surgimento da XML como padrão para troca de informações, a XSLT tornou-se uma linguagem essencial no desenvolvimento de aplicações WEB dinâmicas (FUNG, 2001).

Combinando XSLT com Javascript, pode-se alterar um documento dinamicamente, buscar informações do servidor, construir o formato de apresentação e mostrar o resultado final em uma página HTML, utilizando assim os recursos do AJAX.

XMLHttpRequest é um objeto que permite à *scripts* requisitarem dados via HTTP, protocolo para transferência de dados, a um servidor remoto sem que, no decorrer deste processo, haja a necessidade de recarregamento de toda a página (DARIE et al., 2006).

Conforme Crane, Pascarello e James (2006), com o objeto XMLHttpRequest é possível a troca de dados entre cliente e servidor WEB assincronamente. Dessa forma as requisições realizadas a um servidor WEB podem ser feitas sem a suspensão das atividades do usuário, durante o processo de envio e recebimento de dados, ou seja, a tela do usuário não congela ou trava nos momentos de requisições.

De acordo com Darie et al. (2006) o objeto XMLHttpRequest foi implementado originalmente no *browser* da Microsoft, o Internet Explorer 5.0, com o nome XMLHTTP, através de um controle ActiveX. Atualmente esse objeto é oferecido também por outros *browsers*, como Firefox, Opera e Safari.

Javascript é uma linguagem de criação de *scripts* baseada em objetos e que pode ser incorporada diretamente nas páginas WEB que usam HTML. Quando combinada com DOM definido por um *browser*, permite criar conteúdo em DHTML e aplicativos interativos do lado cliente da WEB. A sintaxe de Javascript baseia-se naquela de linguagens de programação muito utilizadas, como C, C++ e Java (FLANAGAN, 2004).

Autores como (CRANE, PASCARELLO e JAMES, 2006; NIEDERAUER, 2007), definem o Javascript como o personagem central do AJAX, uma vez que todo o processo de uma aplicação AJAX gira em torno do Javascript, pois toda a comunicação entre o usuário e o servidor ocorrerá por meio dessa linguagem.

2.2.3 Princípios do AJAX

Conforme (CRANE, PASCARELLO e JAMES, 2006; NIEDERAUER, 2007) uma aplicação AJAX bem sucedida deve ser projetada seguindo alguns princípios, os quais serão abordados nessa seção.

2.2.3.1 O *browser* hospeda uma aplicação, não conteúdo

Em uma aplicação WEB clássica, baseada em páginas, percebe-se que o *browser* é efetivamente um terminal burro. Ele não sabe nada sobre as ações que o usuário executou até o momento. Essas informações ficam retidas no servidor WEB, tipicamente na sessão do usuário.

> Sessões de usuários no lado servidor são comuns atualmente. Se você está trabalhando em PHP, Java, .NET, Ruby on Rails ou outra linguagem usada para aplicações web, a sessão no lado servidor faz parte da API padrão, assim como controle de solicitações, respostas, e tipos de conteúdo (MIME) (CRANE, PASCARELLO e JAMES, 2006, p. 17) (Tradução nossa).

Conforme Niederauer (2007), quando o usuário entra no site ou inicia uma sessão, vários objetos são criados no lado servidor, representando, por exemplo, a cesta de compras e as credenciais de cliente, em um site de comércio eletrônico. Posteriormente, o servidor WEB envia ao *browser* a página inicial, que inclui códigos HTML, dados do usuário, conteúdos do site e instruções de formatação.

Dessa forma, segundo Crane, Pascarello e James (2006), toda vez que o usuário interage com o site, o *browser* envia uma requisição ao servidor, que retorna um outro documento, contendo a mesma mistura de cabeçalho e dados. Conseqüentemente, o *browser* retira o documento anterior e exibe o novo, mesmo que os dois documentos sejam muito semelhantes. Quando o usuário efetua a saída ou fecha o *browser*, a aplicação é finalizada e a sessão destruída. A figura 2.5 ilustra o ciclo de vida típico de uma aplicação WEB clássica.

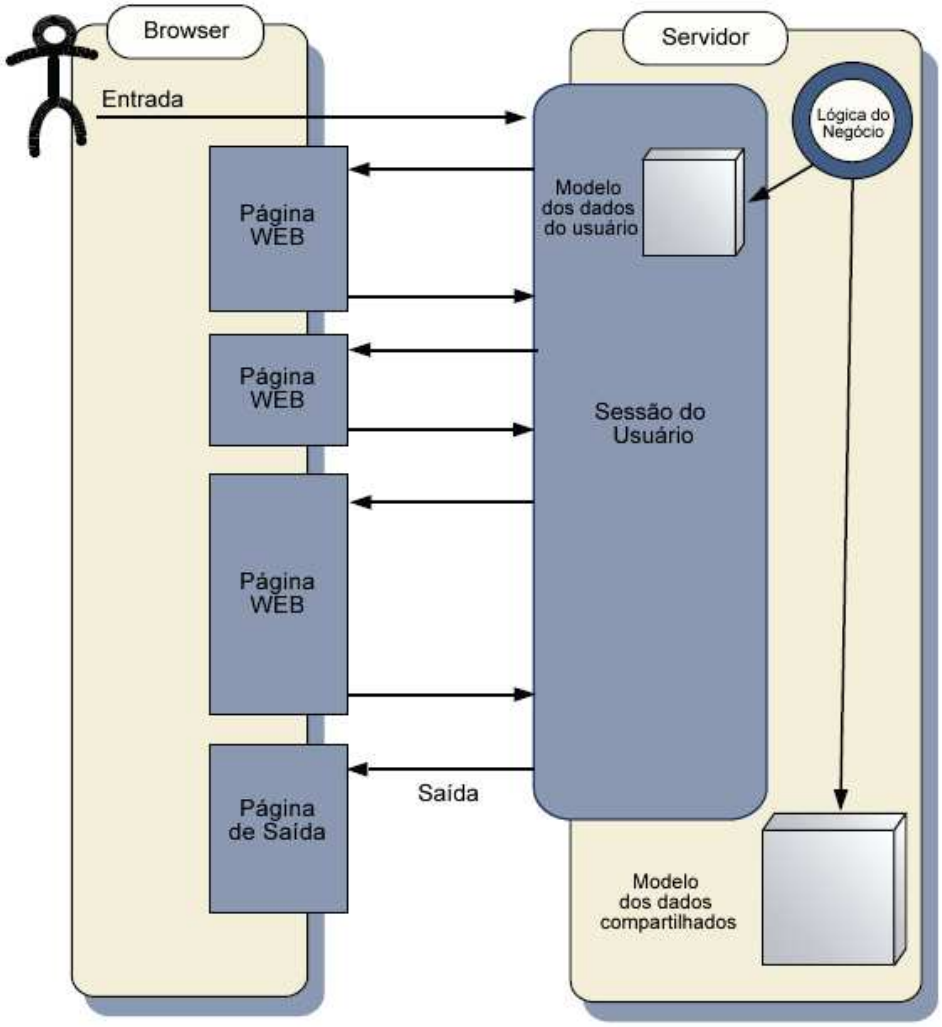


Figura 2.5 - Ciclo de vida típico de uma aplicação WEB clássica.
Fonte: Adaptação do autor segundo (CRANE, PASCARELLO e JAMES, 2006).

No caso de uma aplicação AJAX, parte da lógica da aplicação é movida para o *browser*, por meio da linguagem Javascript, conforme mostra a figura 2.6.

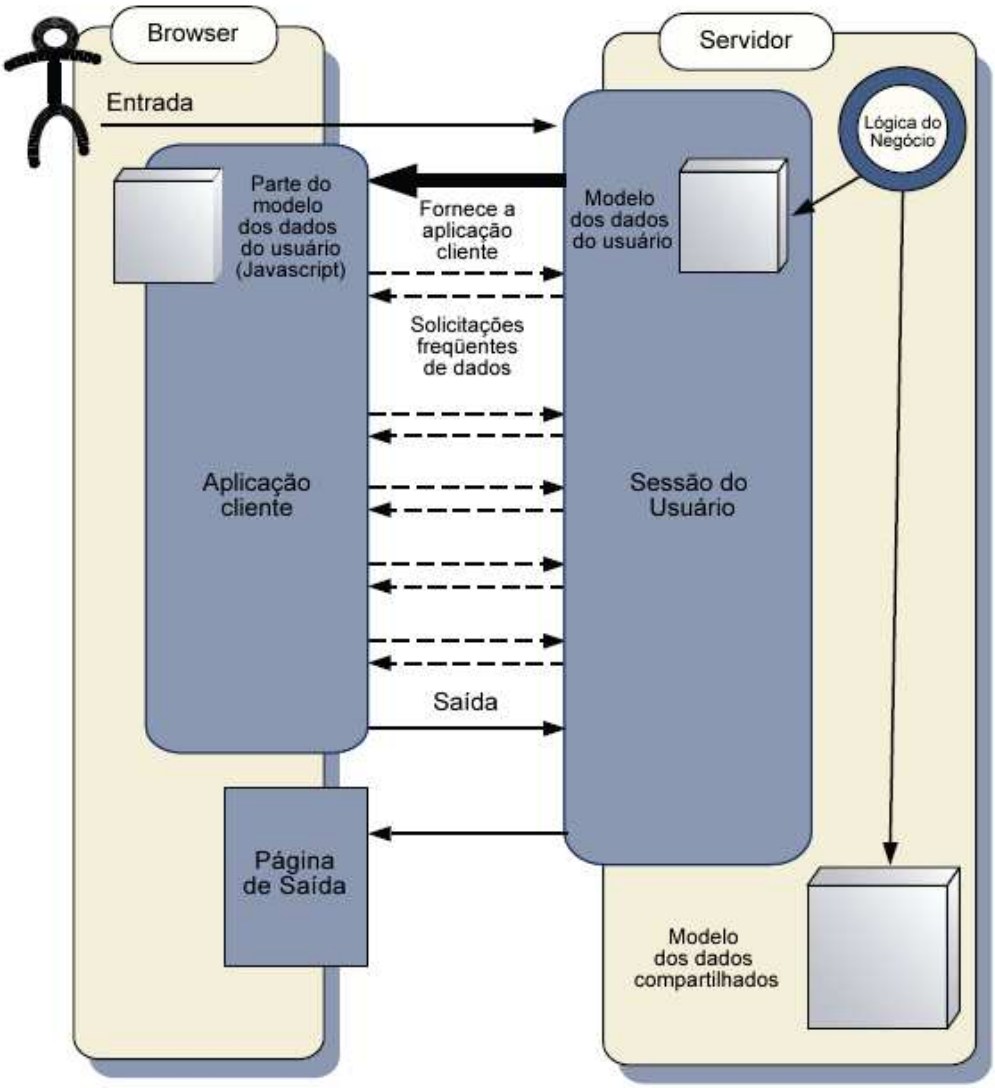


Figura 2.6 - Ciclo de vida de uma aplicação AJAX.
Fonte: Adaptação do autor segundo (CRANE, PASCARELLO e JAMES, 2006).

Neste novo cenário, quando o usuário entra no site, o servidor retorna um documento mais complexo, que será a aplicação do cliente. Diferentemente do modelo clássico, a aplicação do cliente é formada apenas por uma página, que envia requisições freqüentes ao servidor. Essas requisições, que são feitas de maneira assíncrona pelo objeto XMLHttpRequest, podem ser utilizadas para executar diversas operações, como por exemplo, consultas e atualizações nos bancos de dados localizados no servidor (NIEDERAUER, 2007).

2.2.3.2 O servidor fornece dados, e não conteúdo

De acordo com Niederauer (2007), em uma aplicação WEB clássica, a cada requisição que o *browser* faz, o servidor retorna uma página inteira, ou seja, uma mistura de dados do usuário, conteúdo do site e instruções de formatação. Entretanto, nesse novo

modelo, precisamos de uma resposta imediata que contenha apenas as informações que são do nosso interesse, evitando que sejam reenviados todos os conteúdos e códigos que não sofreram qualquer alteração. Por isso se diz que no modelo de aplicação que utiliza AJAX, o servidor deve retornar dados e não conteúdo.

Em uma aplicação AJAX, o tráfego tem sua maior intensidade no início, com um documento complexo sendo entregue quando o usuário entra no site. As comunicações subseqüentes com o servidor são muito mais eficientes, ou seja, à medida que o tempo de interação aumentar, o custo da largura de banda será menor na aplicação AJAX do que na sua aplicação clássica equivalente (CRANE, PASCARELLO e JAMES, 2006), conforme ilustra a figura 2.5.

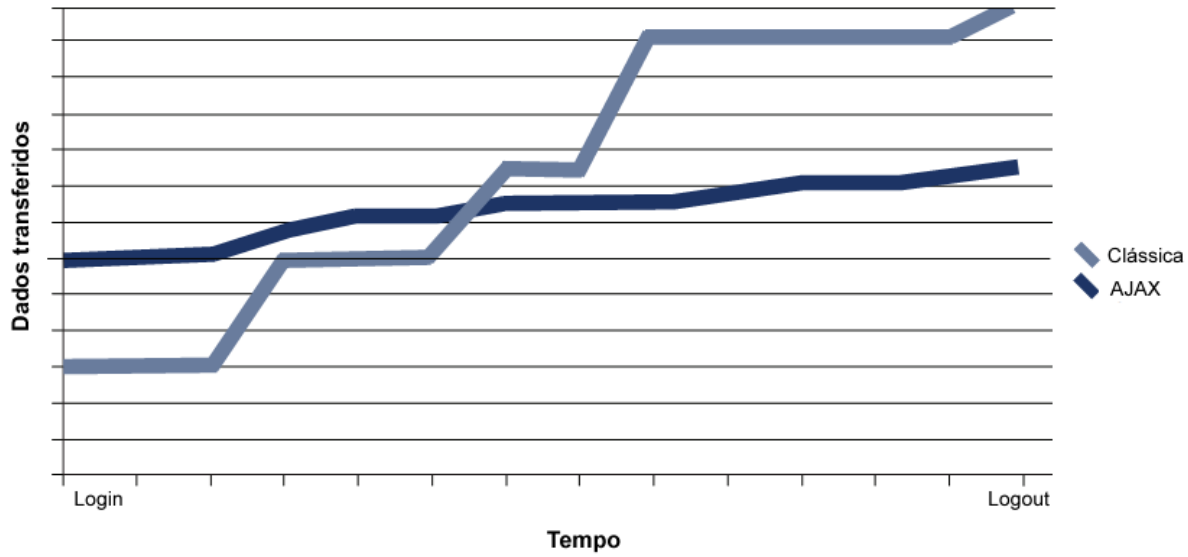


Figura 2.7 - Comparativo de dados transferidos ao longo do tempo.
Fonte: Adaptação do autor segundo (CRANE, PASCARELLO e JAMES, 2006).

2.2.3.3 Interação flexível entre usuário e aplicação

Um *browser* WEB oferece duas maneiras de enviar entradas de dados para um outro computador: *hiperlinks* e formulários HTML. Os *hiperlinks* para proporcionar uma pequena melhoria na interface podem estar vinculados a imagens e folhas de estilo (CSS), como por exemplo, para definir efeitos a serem aplicados quando o mouse estiver sobre eles. Em relação aos formulários HTML, eles nos oferecem um subconjunto básico de componentes de interface com o usuário, como caixas de texto, listas de seleção, botões de rádio e caixas de checagem (*checkboxes*). Entretanto, esses componentes não são suficientes para garantir uma boa interatividade com o usuário. Não existem, por exemplo, tabelas para edição (*grids*),

caixas de combinação ou controles de seleção em árvores como existem nas aplicações *desktop* (CRANE, PASCARELLO e JAMES, 2006).

> A estrutura de uma aplicação web clássica não nos permite chegar sequer próximo do nível de interação de uma aplicação para *desktop*, por exemplo. Ao utilizar JavaScript e folhas de estilo CSS, até conseguimos reproduzir alguns efeitos de interação em um ambiente web, mas mesmo assim a solução ainda pode ser considerada muito rudimentar (NIEDERAUER, 2007, p. 28).

Segundo Niederauer (2007), no caso do AJAX, a interação com o usuário tende a ser mais flexível, contínua e a fluir de forma mais amigável. Não será mais necessário aguardar ao clicar em um *hiperlink* ou submeter um formulário. Conceitos mais sofisticados, como por exemplo, o "arrastar e soltar", tornam-se praticáveis fazendo com que a interface se assemelhe à interface de uma aplicação *desktop*. Dessa forma, torna-se possível combinar a interação do usuário e as solicitações ao servidor de maneira mais completa.

2.2.3.4 Disciplina na codificação

Conforme Niederauer (2007), uma aplicação AJAX é um código Javascript complexo que se comunica com o servidor enquanto o usuário continua trabalhando. Apesar de ser um modelo descendente das aplicações clássicas, as semelhanças entre essas duas formas de programar são pequenas.

Niederauer (2007, p.29) descreve ainda:

> Codificar utilizando AJAX é bastante diferente de codificar uma aplicação clássica para a WEB. Com AJAX, o código fornecido no início da aplicação deve ser executado até que ela seja encerrada, sem interrupções. Para atingirmos esse objetivo, devemos escrever códigos de alto desempenho e de fácil manutenção. Normalmente esses códigos serão muito maiores do que outros escritos em aplicações web clássicas. Portanto, é necessária muita disciplina para desenvolver uma aplicação AJAX.

Atualmente, as aplicações WEB clássicas fazem uso do Javascript apenas em certas ocasiões, devido à limitação que o modelo clássico apresenta. Esse modelo é baseado em páginas e não possibilita que os *scripts* permaneçam ao longo do tempo. Dessa forma, a linguagem Javascript perdeu importância, e muitos desenvolvedores a menosprezam (CRANE, PASCARELLO e JAMES, 2006).

De acordo com Crane, Pascarello e James (2006), nas aplicações AJAX, o Javascript ganha muita importância, pois tende a ser o centro do processo. E como a base do código dessa linguagem passa a ser maior, tornam-se necessárias boas práticas na construção do

mesmo. O código deve tornar-se de preferência, responsabilidade de uma equipe do que apenas de uma pessoa, criando edições de manutenibilidade, separações de interesses, e estilos e padrões de codificação comum. Uma aplicação AJAX, portanto, é uma parte funcional complexa de código que se comunica com o servidor enquanto o usuário continua com seu trabalho.

2.2.4 Características/Vantagens

Especialistas em usabilidade e desenvolvimento WEB elegeram algumas vantagens a serem consideradas em relação à utilização do AJAX nas aplicações que rodam na Internet. Segundo MAHEMOFF (2006), os seguintes benefícios podem ser alcançados:

- **Tráfego mínimo**: As aplicações WEB devem enviar e receber do servidor o mínimo de informação possível. O AJAX pode diminuir consideravelmente o tráfego entre cliente e servidor, pois evita o envio de informação desnecessária. A largura de banda é melhor utilizada, diminuindo a latência de rede, e conseqüentemente obtêm-se um melhor desempenho nas aplicações.

- **Interface amigável**: Aplicações AJAX introduzem um conceito diferente do modelo “clica-espera” suportado atualmente na Internet. Surgem outras funcionalidades do tipo “arrasta e solta” e “duplo clique”. Dessa forma, podem ser desenvolvidas aplicações mais intuitivas, com um conforto visual melhor, além de fornecer um ambiente mais próximo das aplicações *desktop*.

- **Maior interatividade nas aplicações**: Com AJAX não é necessário a utilização de recursos pesados de programação nem de *softwares* proprietários, possibilitando assim disponibilizar aplicações com alta capacidade e usabilidade.

- **Resposta mais rápida**: Em uma aplicação AJAX, os usuários não precisam esperar que a página seja processada por completo cada fez que é feita uma solicitação ao servidor. O usuário pode interagir com uma parte da página enquanto aguarda o retorno, tornando a navegação mais rápida e eficiente.

- **Portabilidade**: Por ser constituído de tecnologias utilizadas pelos principais *browsers* existentes no mercado, o AJAX é uma técnica que não se restringe a um

browser, nem a uma plataforma. Além disso, não requer a instalação de qualquer *plugin* ou *software*.

- **Integração com as linguagens de desenvolvimento WEB**: Outra vantagem é o fato da tecnologia AJAX ser independente de linguagens de programação, oferecendo aos desenvolvedores a possibilidade de escolher a linguagem que se sentem mais à vontade ou a mais apropriada de acordo com o âmbito do projeto.

2.2.5 Desvantagens

Assim como acontece em todas as técnicas e tecnologias utilizadas no desenvolvimento de aplicações WEB, algumas vantagens podem trazer desvantagens. Ao utilizar AJAX, existem alguns pontos negativos a serem considerados:

- **Performance do cliente**: Numa aplicação AJAX transfere-se muito do processamento do servidor para o cliente. Essa mudança tem custos porque estaremos transferindo para o cliente a responsabilidade de realizar determinadas operações para as quais ele não estaria destinado inicialmente. O cliente pode ficar sobrecarregado caso não sejam tomadas algumas precauções durante a fase de desenvolvimento deste tipo de aplicação.

- **Compatibilidade entre *browsers***: Programar em Javascript algumas vezes pode trazer problemas, pois cada *browser* tem suas particularidades. Muitos códigos Javascript que funcionam em um *browser* específico podem ser interpretados de forma diferente em outro, e dessa forma ocasionar alguns erros.

- **Atualização de conteúdo**: A atualização de conteúdo com AJAX pode levar a algumas inconsistências caso não haja um tratamento adequado.

- **Segurança**: Quando são realizados diversos processos pelo lado do cliente, e não no servidor, os mesmos podem gerar alguns riscos de segurança. Ainda não existem padrões muito bem definidos para segurança quando se trata de AJAX.

2.2.6 Ferramentas para desenvolvimento com AJAX

A popularidade do AJAX não pára de aumentar, e conseqüentemente novas bibliotecas e *frameworks* estão surgindo para facilitar a sua utilização, aumentando consideravelmente a produtividade quanto ao tempo de desenvolvimento de forma rápida e simples.

Segundo Minetto (2007), os *frameworks* facilitam o desenvolvimento de *software*, permitindo que os desenvolvedores se ocupem mais com os requerimentos da aplicação do que com os detalhes tediosos, de baixo nível do sistema. Podem incluir coleção de códigos-fonte, classes, funções, técnicas e metodologias que facilitam o desenvolvimento de novas aplicações.

Abaixo serão descritos em linhas gerais algumas bibliotecas e *frameworks* WEB que implementam os recursos do AJAX:

- **AJAXLIB**: Ajaxlib é uma classe *open source* escrita em Javascript que pode ser utilizada em conjunto com várias linguagens WEB, como PHP, PERL e JSP. É uma ferramenta simples e indicada para aplicações que não possuem muitos recursos (AJAXLIB, 2007).

- **DOJO**: Dojo é um *toolkit*[29] DHTML *open source* escrito em Javascript e independente de plataforma. É um dos *frameworks* mais antigos, seu desenvolvimento foi iniciado em 2004 e hoje é considerado um dos produtos mais maduros do mercado (DOJO, 2007).

- **DWR**: *Direct WEB Remoting* (DWR) é um *framework open source* desenvolvido em Java para rodar aplicações AJAX. DWR é um dos mais conceituados *frameworks* AJAX disponíveis para a linguagem Java. A estrutura do código fonte Java fica acessível ao cliente via Javascript, não existindo uma distinção entre lado cliente e servidor, do ponto de vista do desenvolvedor (DWR, 2007).

- **GWT**: *Google Web Toolkit*, ou simplesmente GWT, é um *framework open source* que permite transformar uma aplicação feita em Java em uma aplicação WEB com

[29] *Toolkit* - É um conjunto de elementos básicos para construção de *software*. Normalmente são implementados como uma biblioteca de rotinas ou uma plataforma para aplicativos.

AJAX integrado ao HTML, CSS e Javascript. O GWT foi utilizado para desenvolver os sites Google Maps e Gmail (GWT, 2007).

- **Prototype**: Prototype é um *framework* Javascript que facilita o desenvolvimento de aplicações WEB dinâmicas com AJAX, permitindo o desenvolvimento em classes. Implementa um conjunto de efeitos visuais, como *drag-and-drop*[30]e outros componentes de interface com o usuário (PROTOTYPE, 2007).

- **SAJAX**: SAJAX, acrônimo de *Simple AJAX Toolkit*, é um *framework open source* para AJAX com implementações do lado do servidor. Possui implementações para linguagens como PHP, ASP e ColdFusion (SAJAX, 2007).

- **SYMFONY**: Symfony é um *framework open source* escrito em PHP que segue o paradigma MVC (*Model View Controller*). Tem como principais características a geração de *tableless*[31] e opção de internacionalização (SYMFONY, 2007).

- **Script.aculo.us**: Script.aculo.us é uma biblioteca Javascript que utiliza o *framework* Prototype como base para desenvolvimento. Oferece diversos efeitos extensíveis e suporte para funcionalidades como *drag-and-drop,* auto-completar, entre outros (SCRIPTACULOUS, 2007).

- **XAJAX**: XAJAX é uma classe *open source* em PHP que permite a criação de aplicações WEB em HTML, CSS, Javascript e PHP utilizando AJAX (XAJAX, 2007). Os *frameworks* SAJAX e XAJAX possuem uma diferença básica. No SAJAX o retorno é tratado em uma função Javascript, enquanto no XAJAX o retorno é feito em uma função do PHP, assim como seus comandos e eventos adicionados, resguardando assim o código Javascript.

---

[30] *Drag-and-drop* - Arrastar e soltar elementos em uma página WEB.
[31] *Tableless* - Metodologia utilizada para a concepção do layout de páginasWEB. Maiores informações em: http://www.tableless.com.br

3 PROBLEMAS ENFRENTADOS NA UTILIZAÇÃO DA CLASSE FPDF

A necessidade de uma ferramenta com interface WEB para a criação de relatórios no formato PDF foi o fator motivador desse estudo. Existindo então essa necessidade, o próximo passo foi realizar uma pesquisa bibliográfica com o objetivo de identificar quais bibliotecas ou classes escritas em PHP possuíam a finalidade de geração de arquivos PDF e que conseqüentemente poderiam servir como base para essa nova ferramenta a ser desenvolvida.

Foi escolhida a utilização da classe FPDF para o desenvolvimento da ferramenta, por se tratar de uma classe de simples manuseio que possui inúmeras funcionalidades a serem exploradas, seja de maneira direta ou através de suas extensões. Outro fator de extrema importância para sua escolha é o fato de ser uma classe gratuita que possibilita sua utilização tanto para o uso pessoal quanto comercial, permitindo um estudo mais aprofundado do seu código-fonte e de possíveis modificações conforme a necessidade do desenvolvedor.

A seguir serão apresentadas as principais dificuldades enfrentadas pelos desenvolvedores ao manipular manualmente a classe FPDF na geração de relatórios no formato PDF.

3.1 Montando manualmente um relatório com a classe FPDF

Uma das aplicações mais interessantes que pode ser criada com as funções PDF do PHP é a geração dinâmica de relatórios, baseada em informações obtidas de um banco de dados (NIEDERAUER, 2004). A idéia geral é fazer uma consulta SQL a um banco de dados e gerar um documento PDF com os registros resultantes, em forma de tabela.

A seguir, serão apresentados dois exemplos práticos da criação de relatórios utilizando as funções da classe FPDF, com o objetivo de mostrar as dificuldades enfrentadas

pelos desenvolvedores ao manipular manualmente, no código PHP, a formatação, o posicionamento do texto e demais elementos que formam o relatório.

GERADOR GRÁFICO DE RELATÓRIOS PDF - CATÁLOGO DE LIVROS

TITULO	AUTOR	EDITORA	PÁGINAS	ANO DE PUBLICAÇÃO	IDIOMA
Ajax Design Patterns	Michael Mahemoff	O'Reilly Media	655	2006	Inglês
Ajax in Action	Dave Crane, Eric Pascarello, Darren James	Manning Publication	680	2005	Inglês
Frameworks para Desenvolvimento em PHP	Elton Luis Minetto	Novatec	192	2007	Português
PHP A Bíblia	Tim Converse, Joyce Park	Campus	868	2003	Português
PHP e MYSQL: Desenvolvimento Web	Laura Thomson, Luke Welling	Campus	752	2005	Português
PHP para quem conhece PHP	Juliano Niederauer	Novatec	480	2004	Português
PHP-GTK - Criando Aplicações Gráficas com PHP	Pablo Dall'Oglio	Novatec	445	2007	Português
Professional Ajax	Nicholas C. Zakas, Jeremy McPeak, Joe Fawcett	Wiley	432	2006	Inglês
Web Interativa com Ajax e PHP	Juliano Niederauer	Novatec	288	2007	Português

Figura 3.1 - Exemplo de um relatório PDF gerado a partir de uma consulta SQL.

O primeiro exemplo a ser apresentado consiste na criação de um catálogo de livros, conforme mostra a figura 3.1 acima. Esse relatório será gerado de acordo com uma consulta SQL que foi fornecida, onde serão listados todos os livros cadastrados na base de dados, ordenados pelo título.

A grande vantagem da FPDF é que ela está disponível em forma de uma classe PHP. Sendo assim, não existe a necessidade da instalação e configuração de qualquer extensão no servidor WEB. Basta colocar os arquivos da classe no servidor e inserir uma chamada para ela no início dos programas (NIEDERAUER, 2004).

A partir deste ponto, serão apresentados diversos quadros com o objetivo de gerar o documento PDF visualizado na figura 3.1. Dessa forma, de acordo com o quadro 3.1, o primeiro passo foi definir o diretório onde estão localizadas as fontes da classe (linha 1) e informar a localização do arquivo **fpdf.php** (linha 2), uma vez que nesse arquivo encontram-se as funções utilizadas para a geração do arquivo no formato PDF. Nota-se que nesse exemplo também foram criadas as funções *Header* (linha 4) e *Footer* (linha 11), que tem como objetivo inserir cabeçalho e rodapé nos documentos gerados.

Quadro 3.1 - Definindo os diretórios de localização das fontes e da função FPDF.

```
1.  define('FPDF_FONTPATH','font/');
2.  require('fpdf.php');
3.  class PDF extends FPDF {
4.   function Header() {
5.      $this->SetFont('Arial','B',12);
6.      $this->SetY(10);
7.      $this->Cell(23);
8.      $this->Cell(250,10,'GERADOR GRÁFICO DE RELATÓRIOS PDF - CATÁLOGO
9.       DE LIVROS',1,0,'C');
10.    }
11.   function Footer() {
12.      $this->SetY(-15);
13.      $this->SetFont('Arial','',8);
14.      $this->Cell(0,10,'Página '.$this->PageNo(),0,0,'C');
15.   }
16. }
```

Após a definição dos diretórios de localização da classe e das fontes utilizadas, realiza-se uma conexão com o banco de dados MySQL (linhas 1 e 2) e é executada a consulta que foi definida na variável *$catalogo,* conforme mostra o quadro 3.2 a seguir. Executados esses comandos, criou-se então uma variável chamada *$resultado,* que contém um ponteiro para o conjunto de registros retornados pela consulta. E por fim, a variável *$registros_pagina* define o número de registros que são mostrados por página.

Quadro 3.2 - Conexão com o banco de dados MySQL e execução da consulta SQL.

```
1. include ("mysql_connect.php");
2. include ("mysql_execute.php");
3. $catalogo = "select * from livros order by titulo";
4. $resultado = mysqlexecute($id,$catalogo);
5. $registros_pagina = 25;
```

Finalizada a consulta na base de dados, o próximo passo é criar o documento PDF e definir as configurações gerais do arquivo, de acordo com quadro 3.3. Definidas as configurações gerais do documento, são criadas as variáveis *$posição_horizontal*, para definir

a distância horizontal - em pontos[32] - do texto da tabela até a borda da página, e *$altura_linha* para especificar a altura da linha da tabela.

Quadro 3.3 - Criação e definição das configurações gerais do documento.

```
1. $pdf=new PDF('L','mm','A4');
2. $pdf->Open();
3. $pdf->AddPage();
4. $pdf->SetMargins(2,2,2);
5. $pdf->SetAutoPageBreak(false);
6. $pdf->SetAuthor('Igor Henrique Berlitz');
7. $pdf->SetTitle('Catálogo de Livros');
8. $posicao_horizontal = 25;
9. $altura_linha = 4;
```

Os principais métodos utilizados na configuração do documento são descritos na tabela 3.1 a seguir.

Tabela 3.1 - Métodos utilizados nas configurações gerais do documento PDF.

Método	**Descrição**
Open	Inicia um novo documento PDF.
AddPage	Adiciona uma página ao documento.
SetMargins	Define as margens esquerda, superior e direita.
SetAutoPageBreak	Habilita ou desabilita o modo de quebra automática de página.
SetAutor	Define o autor do documento.
SetTitle	Define o título do documento.

Posteriormente foi criado o cabeçalho da tabela. São definidas a fonte, através do método *SetFont* e a cor de fundo do cabeçalho utilizando a propriedade *SetFillColor*. Para especificar a distância do cursor até as margens horizontal e vertical da página, utilizam-se os métodos *SetY* e *SetX*, respectivamente. O próximo passo foi desenhar as células com os títulos das colunas do relatório. Para isso, o método *Cell* é o mais adequado. Esse método recebe como parâmetros: largura, altura, texto, definição de borda, a posição corrente, o alinhamento

[32] A FPDF trabalha com pontos, tanto para o tamanho da página como para posicionar as coordenadas em cada página. Para referência, o formato A4 tem aproximadamente 595 por 842 pontos e o formato papel de carta norte-americana (*U.S. letter paper*) tem 612 por 792 pontos (WELLING E THOMSON, 2003).

e a definição se o fundo da célula deve ser preenchido ou transparente. Desenhadas as células, necessita-se então atribuir um novo valor para a variável *$posiçao_horizontal*, pois ela será a responsável pela definição da posição horizontal da próxima linha a ser criada na tabela. Todo esse trecho de código está exemplificado no quadro 3.4.

Quadro 3.4 - Definição do cabeçalho da tabela.

```
1.  $pdf->SetFont('Arial','B',7);
2.  $pdf->SetFillColor(232,232,232);
3.  $pdf->SetY($posicao_horizontal);
4.  $pdf->SetX(25);
5.  $pdf->Cell(70,$altura_linha,'TITULO',1,0,'L',1);
6.  $pdf->Cell(60,$altura_linha,'AUTOR',1,0,'L',1);
7.  $pdf->Cell(40,$altura_linha,'EDITORA',1,0,'L',1);
8.  $pdf->Cell(20,$altura_linha,'PÁGINAS',1,0,'R',1);
9.  $pdf->Cell(30,$altura_linha,'ANO DE PUBLICAÇÃO',1,0,'R',1);
10. $pdf->Cell(30,$altura_linha,'IDIOMA',1,0,'L',1);
11. $posicao_horizontal = $posicao_horizontal + $altura_linha;
```

O quadro 3.5 a seguir, apresenta o detalhamento do processo de criação das linhas com a listagem dos livros. Utilizando o comando *while* do PHP (linha 2), foi criado um laço para percorrer os registros. No início do código foi definido o número de registros por página através da variável *$registros_paginas*. Essa variável será utilizada agora, para verificar se existe a necessidade da criação de uma nova página. Caso isso seja necessário, o processo de criação do cabeçalho da tabela será feito novamente.

Ainda dentro do laço criado, foram definidos quais os campos da tabela consultada no banco de dados serão mostrados no relatório. Nesse caso, foram mostrados os seguintes campos: título, autor, editora, número de páginas, ano de publicação e idioma. Após isso, foi especificado o posicionamento horizontal e vertical do cursor utilizando novamente os métodos *SetY* (linha 24) e *SetX* (linha 25). E para escrever os dados nas colunas, mais uma vez o método *Cell* foi executado. A cada linha escrita, a variável *$posicao_horizontal* é incrementada com base na altura da linha definida anteriormente.

Quadro 3.5 - Geração das linhas com os registros do relatório.

```
1.  $i = 0;
2.  while($row = mysql_fetch_array($resultado)) {
3.    $pdf->SetFont('Arial','',7);
4.    $pdf->SetFillColor(255,255,255);
5.     if ($i == $registros_pagina) {
6.       $pdf->AddPage();
7.       $pdf->SetY($posicao_horizontal);
8.       $pdf->SetX(25);
9.       $pdf->Cell(70,$altura_linha,'TITULO',1,0,'C',1);
10.      $pdf->Cell(60,$altura_linha,'AUTOR',1,0,'C',1);
11.      $pdf->Cell(40,$altura_linha,'EDITORA',1,0,'C',1);
12.      $pdf->Cell(20,$altura_linha,'PÁGINAS',1,0,'C',1);
13.      $pdf->Cell(30,$altura_linha,'ANO DE PUBLICAÇÃO',1,0,'C',1);
14.      $pdf->Cell(30,$altura_linha,'IDIOMA',1,0,'C',1);
15.      $posicao_horizontal = $posicao_horizontal + $altura_linha;
16.      $i = 0;
17.    }
18. $titulo  = $row['titulo'];
19. $autor   = $row['autor'];
20. $editora = $row['editora'];
21. $paginas = $row['paginas'];
22. $ano     = $row['ano'];
23. $idioma  = $row['idioma'];
24. $pdf->SetY($posicao_horizontal);
25. $pdf->SetX(25);
26. $pdf->Cell(70,$altura_linha,$titulo,1,0,'L',1);
27. $pdf->Cell(60,$altura_linha,$autor,1,0,'L',1);
28. $pdf->Cell(40,$altura_linha,$editora,1,0,'L',1);
29. $pdf->Cell(20,$altura_linha,$paginas,1,0,'R',1);
30. $pdf->Cell(30,$altura_linha,$ano,1,0,'R',1);
31. $pdf->Cell(30,$altura_linha,$idioma,1,0,'L',1);
32. $posicao_horizontal = $posicao_horizontal + $altura_linha;
33. $i = $i + 1;
34. }
```

Por fim, utilizando o método *Output*, conforme mostra o quadro 3.6, o arquivo PDF será criado. Caso o navegador tenha o *plugin* Adobe Acrobat Reader instalado, o documento

poderá ser visualizado no próprio navegador. Caso contrário será realizado o *download* do PDF gerado.

Quadro 3.6 - Geração do relatório no formato PDF.

```
1. $pdf->Output();
```

Finalizado todo o processo, de criação, formatação e geração do relatório, imagina-se agora o seguinte cenário. Supondo que haja a necessidade de alterar a ordem ou simplesmente excluir uma tabela do relatório. Isso significa reconstruir praticamente todo o relatório, uma vez que a FPDF trabalha com pontos, alterar a coordenada de uma célula representa alterar as propriedades de todas as outras células que compõem o relatório. Sendo assim, fica evidente que por mais simples que seja o relatório, manipular manualmente o código FPDF é uma tarefa trabalhosa, árdua e pouco compensadora.

Com o intuito de justificar a importância da criação de uma ferramenta gráfica que auxilie na geração de relatórios PDF, foi criado um modelo de relatório mais complexo, conforme mostra a figura 3.2.

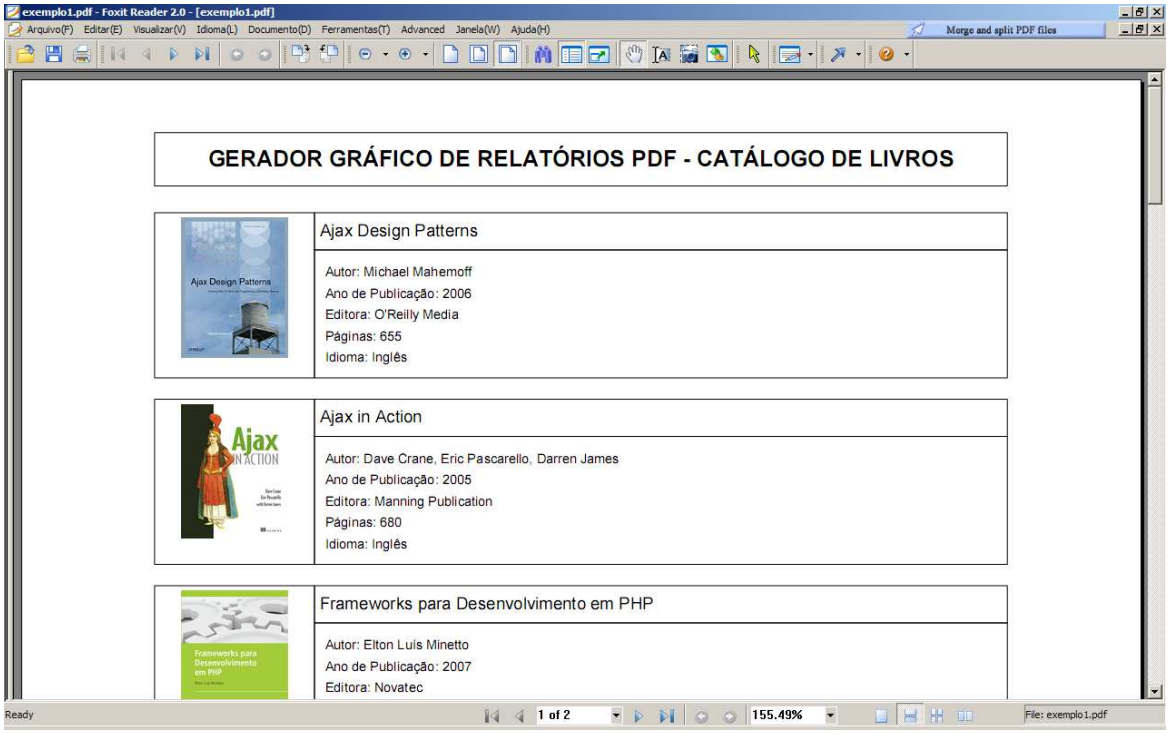


Figura 3.2 - Exemplo de um relatório PDF gerado a partir de uma consulta SQL.

Nota-se que neste relatório as informações estão agrupadas por livro, além de apresentar a imagem do mesmo, o que pode tornar a tarefa mais trabalhosa ainda. Por se tratar de um modelo de relatório mais elaborado, tem-se a necessidade da criação de muitas

variáveis de controle de posicionamento, pois basicamente é desenhada uma célula para cada informação apresentada.

O quadro 3.7 a seguir, indicará o trecho de código onde é definido o posicionamento inicial dos objetos da página, com o objetivo de mostrar a dificuldade enfrentada pelos desenvolvedores na implementação de aplicações desse tipo. O código completo da implementação do relatório será mostrado no Anexo 1.

Quadro 3.7 - Posicionamento inicial dos objetos da página.

```
1.  //Tamanho da célula da imagem
2.  $altura_celula_img = 30;
3.  $largura_celula_img = 31;
4.  //Posicionamento da célula da imagem
5.  $pos_horizontal_cel_img = 25;
6.  $pos_vertical_cel_img = 25;
7.  //Posicionamento da imagem
8.  $posicao_horizontal_img = $pos_horizontal_cel_img + 1;
9.  $posicao_vertical_img = $pos_vertical_cel_img + 5;
10. //Posicionamento do titulo
11. $pos_horizontal_titulo = 25;
12. $posicao_vertical_titulo = 55;
13. //Posicionamento da célula dos dados
14. $pos_horizontal_info = 55;
15. $pos_vertical_info = 32;
16. //Posicionamento do texto
17. $pos_horizontal_autor = 37;
18. $pos_horizontal_ano = 41;
19. $pos_horizontal_editora = 45;
20. $pos_horizontal_paginas = 49;
21. $pos_horizontal_idioma  = 53;
22. $pos_vertical_texto = 57;
23. //Espaçamento entre linhas
24. $espaco_linha = 35;
```

Definido o posicionamento inicial dos objetos da página, o próximo passo é criar um laço que percorra os registros consultados. Dentro desse laço são listadas as informações do livro e conseqüentemente os valores das variáveis de posicionamento são incrementados levando em consideração a posição atual do objeto, seja ela vertical ou horizontal, e o espaçamento entre linhas definido. O quadro 3.8 mostra o trecho de código descrito acima.

Quadro 3.8 - Atualizando valor das variáveis de posicionamento.

```
1. $pos_horizontal_cel_img = $pos_horizontal_cel_img + $espaco_linha;
2. $posicao_horizontal_img = $posicao_horizontal_img + $espaco_linha;
3. $pos_horizontal_titulo  = $pos_horizontal_titulo + $espaco_linha;
4. $pos_vertical_info      = $pos_vertical_info + $espaco_linha;
5. $pos_horizontal_autor   = $pos_horizontal_autor + $espaco_linha;
6. $pos_horizontal_ano     = $pos_horizontal_ano + $espaco_linha;
7. $pos_horizontal_editora = $pos_horizontal_editora + $espaco_linha;
8. $pos_horizontal_paginas = $pos_horizontal_paginas + $espaco_linha;
9. $pos_horizontal_idioma  = $pos_horizontal_idioma + $espaco_linha;
```

Supondo que haja a necessidade da inclusão de mais uma coluna, entre a foto do livro e seus dados, e uma linha antes de “Ano de Publicação”, seria necessário refazer todo o processo de posicionamento das informações, pois ao alterar a coordenada de uma célula, as propriedades das outras células que compõem o relatório também terão que ser alteradas.

Diante desse cenário, ficou clara a importância da criação de uma ferramenta gráfica que auxilie os desenvolvedores na criação de relatórios no formato PDF, pois a manipulação manual do código PHP pode ser tornar complexa e cansativa, na necessidade da alteração, exclusão ou inserção de um campo no relatório.

4 EASYFPDF: CRIANDO RELATÓRIOS PDF

Com base em informações sobre a melhor forma de desenvolvimento de aplicações que rodam na Internet e análise dos principais problemas encontrados na utilização da classe FPDF, esta seção se propõe a apresentar, de forma insipiente, a estrutura de uma ferramenta que possibilite ao desenvolvedor criar os relatórios a serem exportados em PDF, através de uma interface WEB simples e intuitiva.

4.1 Tecnologias a serem utilizadas

Nesta seção, serão abordadas as tecnologias utilizadas para o desenvolvimento da ferramenta, apresentando suas características principais e seu papel na construção do sistema.

4.1.1 PHP

O PHP é uma linguagem que permite criar aplicações WEB dinâmicas, possibilitando uma interação com o usuário através de formulários, parâmetros de entrada, entre outras características. Os motivos para a escolha do PHP como linguagem de programação no desenvolvimento da ferramenta são inúmeros, e podem ser explicados pelas suas próprias características: eficiência, segurança, código-fonte aberto, estrutura simples, multiplataforma, conectividade com vários bancos de dados e facilidade no aprendizado.

4.1.2 MYSQL

O MySQL é um SGBD (Sistema de Gerenciamento de Banco de Dados Relacional) que utiliza a linguagem padrão SQL (*Structured Query Language*) e é largamente utilizado em aplicações WEB. O MySQL é o mais popular SGBD *open-source*, podendo ser distribuído livremente e disponibiliza seu código fonte para *download* (MYSQL, 2007).

De acordo com Prates e Niederauer (2006) o MySQL é uma alternativa atrativa porque mesmo possuindo uma tecnologia complexa de banco de dados, seu custo é bastante baixo. Têm como destaque suas características de velocidade, escalabilidade e confiabilidade, fatores que vem fazendo com que seja adotado por departamentos de TI e desenvolvedores WEB.

Nesse contexto, o MySQL foi escolhido na fase inicial de desenvolvimento da ferramenta, por se tratar de um SGBD rápido e de alta performance, principalmente em um ambiente WEB. Além de rápido, o MySQL pode ser facilmente conectado a aplicações PHP e possui ferramentas de backup, controle de usuários e acessos, verificação e correção de tabelas corrompidas.

4.1.3 ADODB

ADODB é uma biblioteca *open-source* para abstração de dados escrita em PHP. Dentre os principais bancos de dados suportados pela biblioteca, podemos citar: MySQL, Oracle, Microsoft SQL Server, Sybase, PostgreSQL, Interbase, Foxpro e Access (ADODB, 2007).

Segundo Niederauer (2004) a abstração de banco de dados é um recurso extremamente útil, pois possibilita a criação de aplicações portáteis entre diferentes SGBDs. O mesmo programa poderá ser utilizado no MySQL, PostgreSQL, Oracle e todos os outros bancos suportados pela biblioteca que está sendo utilizada, conforme ilustra a figura 4.1.

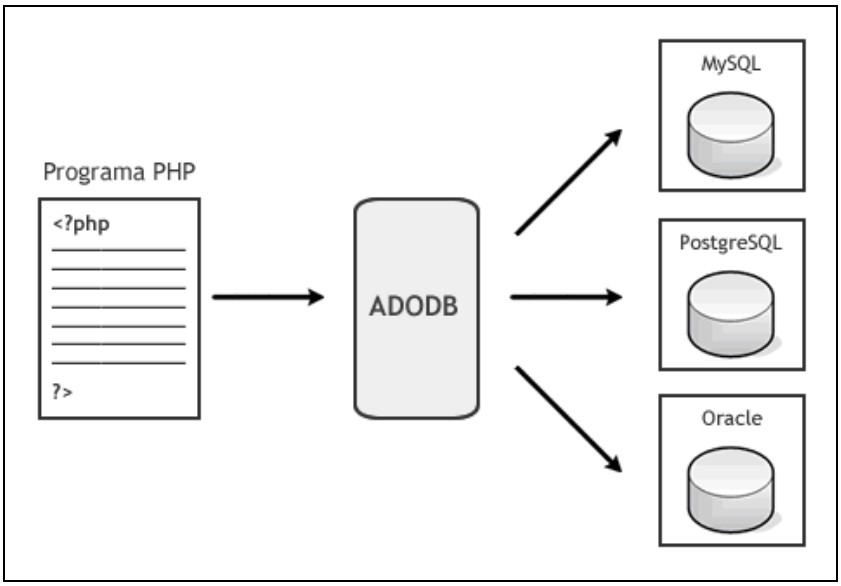


Figura 4.1 - Interação da ADODB com os SGBDs.
Fonte: Adaptação do autor segundo (NIEDERAUER, 2004).

4.1.4 XHTML e CSS

A ferramenta foi desenvolvida sob a plataforma WEB, com o objetivo de oferecer uma interface intuitiva e de fácil manipulação por parte do usuário, permitindo assim a configuração do relatório de forma simples e ágil. Essa interface foi implementada seguindo os padrões recomendados pelo W3C, utilizando as linguagens XHTML e CSS. Optou-se por utilizar essas linguagens, pois além de permitirem um maior nível de padronização, oferecem páginas WEB mais leves e semanticamente corretas.

4.1.5 AJAX e Javascript

Outro ponto a ser destacado no que diz respeito à interface com o usuário, é a utilização da tecnologia AJAX que permite uma interação mais dinâmica entre a aplicação e o usuário, além de reduzir a carga de trabalho sobre o servidor. A idéia assim é proporcionar uma navegação atrativa e rápida sem aqueles cansativos e intermináveis tempos de espera após um clique.

Fez-se o uso de bibliotecas Javascript auxiliares, com o intuito de facilitar a utilização do AJAX e aumentar a produtividade quanto ao tempo de desenvolvimento. O *framework* Prototype, explicado na seção 2.2.6 deste trabalho, assim como a biblioteca Script.aculo.us., foram escolhidos por disponibilizarem recursos e efeitos que até pouco tempo não eram muito utilizados em aplicações WEB: *drag-and-drop* (arrastar e soltar), *slider* (escorregar), redimensionar, auto-completar, e janela modal[33].

4.1.6 XML

XML, acrônimo de *Extensible Markup Language*, pode ser definida como uma meta linguagem de marcação criada com o objetivo de trocar e/ou transportar dados de uma aplicação para outra, integrando sistemas de informação. Por meio de uma estrutura de documentos utilizando *tags*, a XML possibilita ao usuário a criação de uma linguagem de marcação específica, de acordo com a sua necessidade (SILVA FILHO, 2004).

---

[33] Janela modal é um elemento que bloqueia qualquer interação na janela principal até que a tarefa que está sendo realizada seja encerrada.

Conforme Ray (2001) a linguagem XML foi implementada seguindo as premissas de fornecer uma linguagem orientada por conteúdo, extensível, derivada de SGML (*Standard Generalized Markup Language*), independente da plataforma e com capacidade ilimitada para adaptar-se a uma grande variedade de aplicações.

A estrutura de um documento XML é dada com o uso de marcação, que consiste em elementos. Por sua vez, um elemento XML consiste em uma *tag* inicial e uma *tag* final, exceto no caso de elementos que são definidos para serem vazios, que consistem em apenas uma *tag* (DAUM e MERTEN, 2002), conforme mostra o exemplo no quadro 4.1 a seguir.

Quadro 4.1 Exemplo da definição de elementos XML.

Documento XML com elementos que possuem tags de início e fim

```
<?xml version="1.0">
<nome>
     <primeiro>Rich</primeiro>
     <ultimo>Fremont</ultimo>
     <apelido>Vinnie</apelido>
</nome>
```

Documento XML com elementos vazios

```
<?xml version="1.0">
<nome apelido="Vinnie"/>
```

Fonte: Adaptação do autor conforme (CHOI et al., 2001).

Cada elemento possui um tipo, identificado por um nome, e que pode ter um conjunto de especificações de atributos. O uso de atributos permite descrever detalhes sobre o elemento mais claramente. Um atributo pode ser usado para dar ao elemento um rótulo exclusivo, de modo que possa ser facilmente localizado, ou pode descrever uma propriedade sobre um elemento. Um elemento pode ter qualquer quantidade de atributos, desde que cada um tenha um nome diferente (RAY, 2001). Para um melhor entendimento, o quadro 4.2 apresenta um exemplo da utilização de atributos em um elemento XML.

Quadro 4.2 Exemplo da utilização de atributos em um elemento XML.

```
<?xml version="1.0">
<nome apelido="Vinnie">
     <primeiro>Rich</primeiro>
     <ultimo>Fremont</ultimo>
</nome>
```

Fonte: Adaptação do autor conforme (CHOI et al., 2001).

De acordo com Choi et al. (2001) a escolha entre atributos e elementos geralmente é dirigida por uma preferência pessoal. Em alguns casos, pode fazer mais sentido usar uma forma ou outra para obter uma estrutura consistente para os seus documentos.

De acordo com Daum e Merten (2002) uma das grandes vantagens na utilização do XML é a extensa relação de bibliotecas e APIs (*Application Programming Interface*) que as linguagens de programação vem oferecendo para análise (*parser*) e manipulação de documentos XML. Para a realização do *parser* do arquivo XML de configuração da ferramenta descrita nesse capítulo, utilizou-se a classe KXParse que possibilita a análise sintática de documentos XML de forma rápida, simples e eficiente.

4.1.7 KXParse

KXParse é uma classe *open-source* que permite manipular arquivos XML sem a necessidade de utilizar extensões nativas do PHP. Possibilita ao desenvolvedor criar, editar e analisar documentos XML de forma simples e objetiva, manipulando-os através de um mapeamento de *tags*.

Para fazer uso da KXParse o desenvolvedor precisa apenas copiar os arquivos da classe para o servidor WEB e inserir uma chamada no início dos programas, sem a necessidade de instalar e configurar qualquer outra extensão no PHP (KXParse, 2007).

O acesso à estrutura do documento XML com a utilização da KXParse é feita através de árvores que contém nós que levam para todas as partes deste documento. As principais capacidades fornecidas pela KXParse são as seguintes:

- Localizar o nó raiz em um arquivo XML;
- Localizar uma lista de elementos com um determinado nome de *tag*;
- Obter uma lista de filhos de um determinado nó;
- Obter o elemento pai de um determinado nó;
- Inserir, alterar ou remover um elemento ou atributo no arquivo XML.

Na seção 4.4.3, a seguir, será apresentado um exemplo explicando detalhadamente o modo de utilização da classe.

4.2 Funcionalidades

Visando maior clareza e simplicidade, a seguir serão listadas as funcionalidades implementadas na ferramenta com o objetivo de atender as necessidades dos desenvolvedores na geração de arquivos PDF:

- Permite ao desenvolvedor escolher um *template* para apresentação dos dados;
- Possibilita a definição do tipo de página (retrato ou paisagem) e tamanho das margens;
- Proporciona a personalização das informações a serem mostradas no cabeçalho e rodapé do relatório, como por exemplo, título, subtítulo e numeração da página;
- Permite definir o formato, tipo e cor da fonte do texto a ser inserido no documento;
- Possibilita arrastar e soltar os campos que serão apresentados no relatório;
- Permite alterar a largura e o nome dos campos que compõem o relatório;
- Possibilita uma pré-visualização do relatório a ser gerado.

4.3 Arquitetura

Apresentadas as tecnologias utilizadas no desenvolvimento da solução e as funcionalidades em linhas gerais, essa seção tem como objetivo explicar os aspectos principais da arquitetura da EasyFPDF, apresentando o fluxo básico de funcionamento do sistema e a forma de interação entre as tecnologias envolvidas.

A figura 4.2 ilustra um modelo de navegação interna na Internet, utilizando o navegador, servidor WEB, mecanismos de *script* e servidor de banco de dados. Todo o fluxo de funcionamento será descrito a seguir.

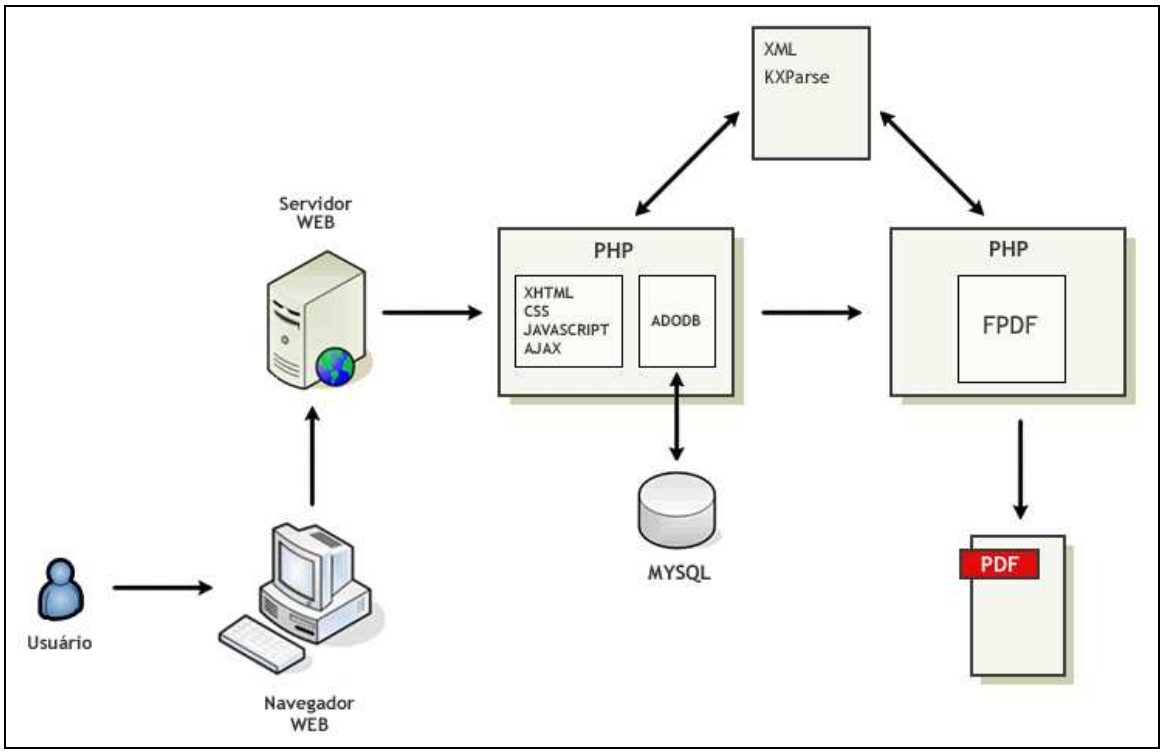


Figura 4.2 - Fluxo da Arquitetura.

(1) O usuário faz uma solicitação ao navegador, esta solicitação é enviada ao servidor WEB através do protocolo *http*;

(2) O servidor WEB recebe essa solicitação e envia para o mecanismo de *script*, onde estão localizadas as linguagens XHTML, CSS, Javascript e a técnica AJAX;

(3) O mecanismo de *script* processa a solicitação realizada pelo usuário e encaminha para o código PHP onde está localizada a biblioteca ADODB, que por sua vez encaminha para o servidor de banco de dados (MySQL);

(4) O servidor de banco de dados (MySQL) lê a informação e devolve para o mecanismo de *script*;

(5) O mecanismo de *script* termina o processamento, formata a solicitação e devolve para o servidor WEB onde se encontra a classe KXParse, responsável pela configuração do relatório através de um arquivo XML;

(6) A KXParse manipula o documento XML, e em conjunto com a FPDF, monta o relatório e devolve para o usuário, o qual visualiza na tela o relatório no formato PDF.

4.4 Geração de relatórios utilizando a ferramenta EasyFPDF

A ferramenta EasyFPDF, tema deste trabalho, tem a finalidade de gerar relatórios no formato PDF, a partir da coleta de dados contidos no banco de dados da aplicação, através de uma interface WEB amigável e de simples manuseio, facilitando assim a manipulação das rotinas de programação por parte do desenvolvedor.

4.4.1 Organização do relatório

Baseado em Villas e Villasboas (1988), a configuração do relatório a ser gerado pela ferramenta EasyFPDF está organizado em três níveis de controle: nível de relatório, nível de página e nível de campo. Estes níveis serão detalhados nas subseções a seguir.

4.4.1.1 Nível de Relatório

Neste nível são tratadas as informações gerais sobre o relatório a ser gerado. Divide-se em três partes:

- **Cabeçalho**: O cabeçalho corresponde à identificação do sistema, ou seja, possibilita ao desenvolvedor apresentar as informações gerais do relatório, tais como: título, subtítulo, dados do autor, data de criação e número de páginas.

- **Corpo**: O corpo deste nível corresponde às páginas que compõem o relatório.

- **Rodapé**: O rodapé corresponde à área do relatório onde poderão ser inseridas informações gerais da página.

4.4.1.2 Nível de Página

No nível de página são tratadas as informações que compõe uma página do relatório. São divididos em três partes:

- **Cabeçalho:** O cabeçalho deste nível é a primeira informação que compõe uma página, como por exemplo, o título da página.

- **Corpo:** o corpo deste nível refere-se à área reservada para a apresentação das informações que estão dentro dos campos.

- **Rodapé:** O rodapé é a área reservada onde deverão ser inseridos os totais gerais de todos os dados apresentados na página do relatório, e também informações gerais sobre a página.

4.4.1.3 Nível de Campo

Cada informação exibida em uma página do relatório chamamos de campo. Um campo pode conter informação de vários tipos de dados, e está dividido em três partes:

- **Cabeçalho:** O cabeçalho corresponde ao nome do campo.

- **Corpo:** O corpo apresenta as informações que compõem o campo e que devem ser impressas nas páginas.

- **Rodapé:** O rodapé é a área reservada para a apresentação dos totalizadores das informações apresentadas nos campos.

4.4.2 Origem dos dados

Para a construção do relatório faz-se necessário a utilização de um SGBD para que o relatório possa ser preenchido com os dados provenientes dessa base de dados. A ferramenta FPDF foi desenvolvida para dar sustentação a diversos bancos de dados, pois utiliza o conceito de abstração de dados fazendo uso da biblioteca ADODB. O banco de dados *open-source* MySQL foi utilizado na fase inicial de implementação da ferramenta.

Quadro 4.3 Exemplo da utilização da ADODB com o MySQL.

```
<?php
  //Incluindo a biblioteca ADODB no arquivo PHP
  include_once('resources/adodb/adodb.inc.php');
  //Definindo qual a base de dados que será acessada
  $dbName = "TC";
  //Fazendo a chamada da ADODB com o MySQL
  $db = NewADOConnection("mysql");
  //Conectando na base de dados
  $db->Connect("localhost", "root", "admin", "$dbName") or die("Erro ao
conectar o MySQL!");
?>
```

4.4.3 Seções do Relatório

Um relatório é composto por páginas que recebem os dados a serem apresentados. As páginas estão divididas em duas seções: a seção de configuração do relatório e a seção dos campos a serem apresentados.

4.4.3.1 Configuração do relatório

A configuração do relatório é definida através de um arquivo XML, onde são especificados o *template* do relatório e o mapeamento dos dados para os campos dentro do *layout* do relatório. Neste arquivo são configuradas as seguintes opções: formato da página, margens, cabeçalho, rodapé, localização e formatação dos campos. Inicialmente, esse arquivo será carregado com algumas informações pré-definidas, possibilitando ao desenvolvedor alterá-las através de uma interface WEB, que será apresentada em detalhes na próxima seção.

Depois de configurado conforme as preferências do desenvolvedor, esse arquivo passa por um processo de tradução e manipulação com a utilização da classe KXParse. O quadro 4.4 apresenta o código do arquivo XML de configuração do relatório.

Quadro 4.4 Arquivo XML de configuração do relatório.

```
1.  <?xml version="1.0" encoding="UTF-8" ?>
2.  <page>
3.    <pageConfig>
4.      <template type="simples"/>
5.      <orientation format="P"/>
6.      <margin top="2" left="2" right="2" bottom="3"/>
7.    </pageConfig>
8.    <pageHeader>
9.      <title align="center" fontColor="#000000" fontType="arial"
fontSize="12" fontBold="true" fontItalic="false">
10.       Título do Relatório
11.     </title>
12.     <subTitle align="center" fontColor="#000000" fontType="arial"
fontSize="12" fontBold="false" fontItalic="false">
13.       Subtítulo do Relatório
14.     </subTitle>
15.   </pageHeader>
16.   <pageFooter>
17.     <footer align="center" fontColor="#000000" fontType="arial"
fontSize="10" fontBold="false" fontItalic="false">
18.       Rodapé do Relatório
19.     </footer>
20.   </pageFooter>
21.   <fields>
22.   </fields>
23. </page>
```

Como pode ser observado no quadro 4.4 acima, o documento XML gerado para estruturar o relatório abrange não apenas os dados de configurações do relatório, mas também os campos que o compõe. No entanto, por se tratar de um arquivo inicial de configuração, onde os campos que são apresentados ainda não foram definidos pelo desenvolvedor, o elemento <fields> fica vazio. Para um melhor entendimento, a disposição da declaração dos elementos é a seguinte:

- Na linha 2 está situado o elemento <page>, elemento principal do documento XML. A partir desse elemento são criados seus filhos <pageConfig>, <pageHeader>, <pageFooter> e <fields>.

- Entre as linhas 3 e 7 estão declarados os elementos de configuração do relatório, como template, orientação da página e margens (template, orientation e margin).

- Entre as linhas 8 e 15 estão declarados os elementos que compõe o cabeçalho da página.

- Entre as linhas 16 e 19 estão declaradas as propriedades do rodapé da página.

- Por fim, na linha 21 é declarado o elemento <fields> que posteriormente conterá todos os campos do relatório.

Utilizando a estrutura XML apresentada no quadro 4.4, o próximo passo então é realizar o processo de tradução e manipulação desse documento XML através da classe KXParse. O quadro 4.5 ilustra a utilização da KXParse para traduzir as propriedades de configuração do relatório (template, orientação e margens).

Quadro 4.5 – Exemplo de utilização da classe KXParse.

```
1. include_once ("kxparse.php");
2. $xml = new kxparse(getcwd()."/config.xml");
3. echo $xml-> get_attribute("page:config:template","1:1:1","type");
4. echo $xml-> get_attribute("page:config:orientation","1:1:1","format");
5. echo $xml->get_attribute ("page:config:margin","1:1:1","top");?>
6. echo $xml->get_attribute ("page:config:margin","1:1:1","bottom");?>
7. echo $xml->get_attribute ("page:config:margin","1:1:1","left");?>
8. echo $xml->get_attribute ("page:config:margin","1:1:1","right");?>
```

Conforme mostra o quadro 4.5 acima, a KXParse pode ser utilizada de forma simples e objetiva. O primeiro passo é incluir uma chamada para a classe no início do código PHP (linha 1), e em seguida criar uma variável referenciando o arquivo XML a ser manipulado, nesse exemplo denominado *config.xml* (linha 2).

Nota-se que utilizamos o método *get_attribute* e através de uma mapeamento de *tags* escrevemos na tela as seguintes definições do documento XML criado:

- O *template* escolhido (linha 3).

- A orientação da página (linha 4).

- E por fim, o tamanho das margens (linhas 5, 6,7 e 8).

4.4.3.2 Campos que compõem o relatório

Através dos campos definidos dentro do *layout* do relatório é que se podem apresentar os dados provenientes de uma base de dados. Os campos do relatório devem relacionar as colunas correspondentes ao objeto selecionado na tabela do banco de dados, sendo que sua identificação deve conter os nomes da tabela e da coluna referenciadas. Por exemplo, para que os dados da coluna **id_livro** da tabela **Livro** sejam mapeados para o relatório, um campo id_livro deve ser definido no arquivo XML especificado anteriormente no quadro 4.4.

O quadro 4.6, a seguir, mostra como é declarado um campo no arquivo de configuração XML.

Quadro 4.6 Definição das *tags* de campos.

```
<fields>
   <field id=”livros.id_livro” align=”left” bgcolor=”#ffffff” width=”60”>
      Livro
   </field>
</fields>
```

4.5 GUI – Interface com o Usuário

Para a implementação da interface gráfica da aplicação (GUI - *Graphical User Interface*) seguiram-se as recomendações da W3C para acessibilidade do conteúdo WEB (WCAG, 2007). O principal objetivo dessas recomendações é indicar boas práticas na criação e interpretação dos conteúdos para a WEB. Aplicações desenvolvidas seguindo esses padrões podem ser acessadas e visualizadas por qualquer pessoa ou tecnologia, independente de *hardware*, *software* ou plataforma, de maneira rápida e compatível com os novos padrões e tecnologias que possam surgir com a evolução da internet.

Baseado nesse contexto, a ferramenta EasyFPDF foi implementada com o propósito de funcionar corretamente nos principais *browsers* utilizados para acesso a WEB, que segundo W3Schools (2007) são: Internet Explorer 6, Internet Explorer 7, Firefox e Opera.

4.5.1 Apresentando a ferramenta

O processo de geração de um relatório pode ser resumido em duas fases, a definição da aparência, ou seja, o *layout* do relatório, e o mapeamento do banco de dados para os

campos dentro do *layout* definido. Seguindo essa linha de raciocínio, a interface da ferramenta é composta das seguintes partes:

- **Menu Superior**, que contém o logotipo da ferramenta, o item Configurações que permite alterar as propriedades do relatório e o botão de geração do arquivo PDF, conforme o item 1 da figura 4.3;

- Aba lateral **Base de Dados**, onde é apresentado o mapeamento das tabelas do banco de dados, listando os campos a serem incluídos no relatório, vide item 2 da figura 4.3. Na fase inicial da aplicação, o desenvolvedor deverá informar manualmente no código PHP qual a base de dados que será utilizada. Propõe-se como trabalho futuro a criação de uma interface gráfica que possibilite a configuração da mesma.

- Aba principal, chamada **Folha A4**, seção onde será feita a montagem do relatório, de acordo com o item 3 da figura 4.3.

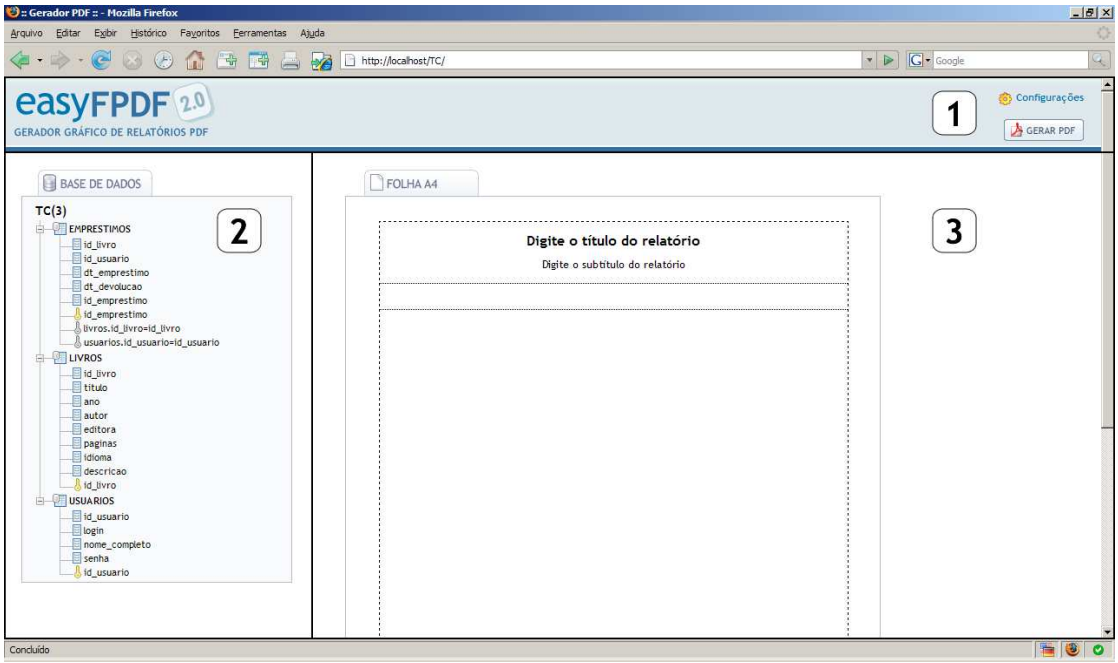


Figura 4.3 – Janela principal da ferramenta.

4.5.2 Configurações do relatório

Conforme explicado anteriormente, a janela principal da ferramenta será carregada com base em algumas informações pré-definidas no arquivo XML de configuração. A alteração dessas informações poderá ser feita através da opção *Configurações*, localizada no menu superior da aplicação.

Ao clicar nesse item, será aberta na tela uma janela modal com as opções disponíveis para a personalização do relatório. Essa janela é composta pelas seguintes abas de navegação:

- ***Templates*:** Definição do *template* que será utilizado na apresentação dos dados do relatório, conforme mostra a figura 4.4;

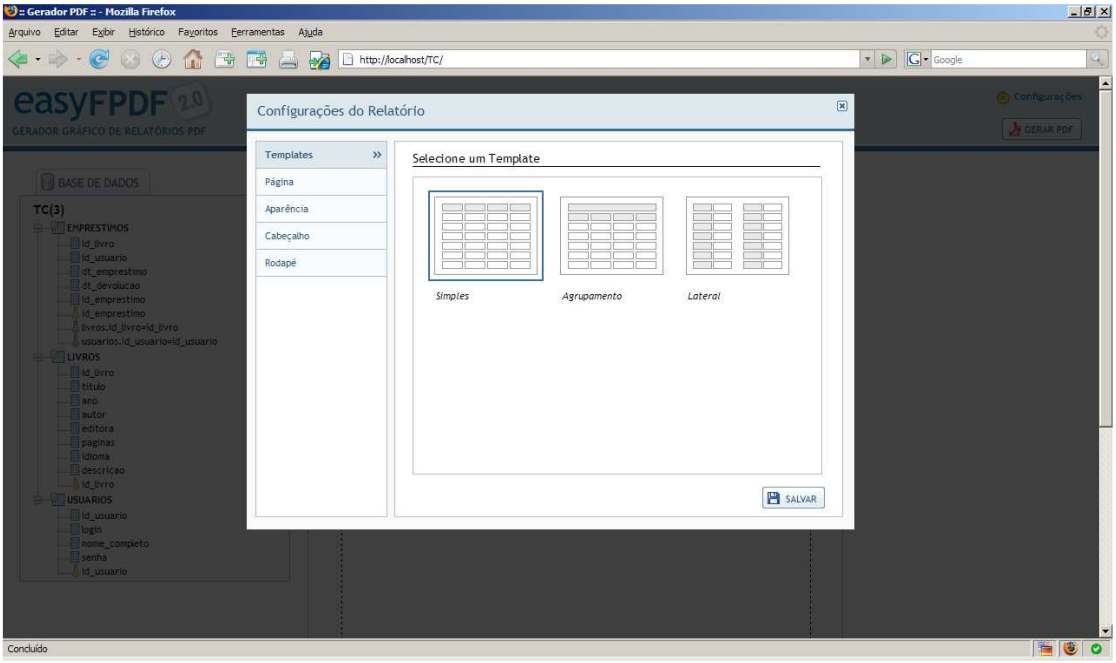


Figura 4.4 – Janela de configuração do template do relatório.

- **Cabeçalho:** Personalização do cabeçalho do relatório;

- **Rodapé**: Personalização do rodapé do relatório.

- **Página:** Seleção do formato da página e definição do tamanho das margens. A figura 4.5 mostra a tela;

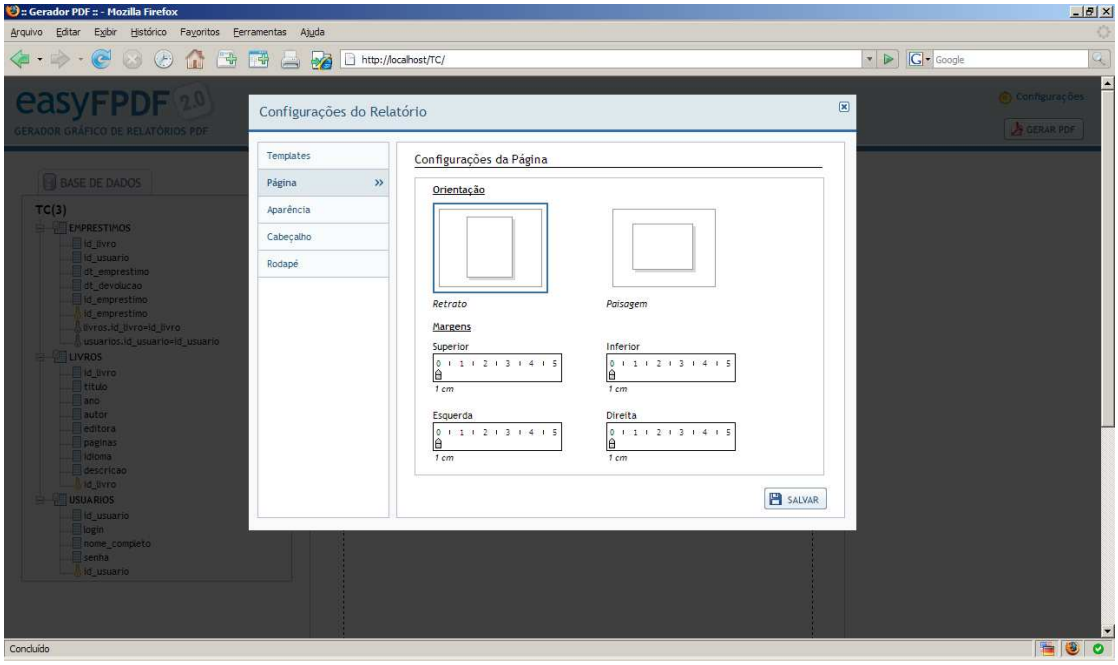


Figura 4.5 – Janela de configuração da página do relatório.

- **Aparência:** Personalização do tamanho, cor, fonte dos textos e dos demais itens do relatório;

4.5.3 Mapeando a base de dados

A aba lateral Base de Dados, tem como objetivo principal listar todas as tabelas provenientes do banco de dados utilizado. Com o intuito de proporcionar uma navegação amigável e de fácil manipulação por parte do desenvolvedor, fez-se o uso da biblioteca ADODB em conjunto com a linguagem Javascript, para criar um menu no estilo *treeview*[34], de modo que cada item da tabela possa ser identificado, conforme mostra a figura 4.6.

[34] *Treeview*: É um controle em estilo de árvore muito utilizado para exibir dados de natureza hierárquica.

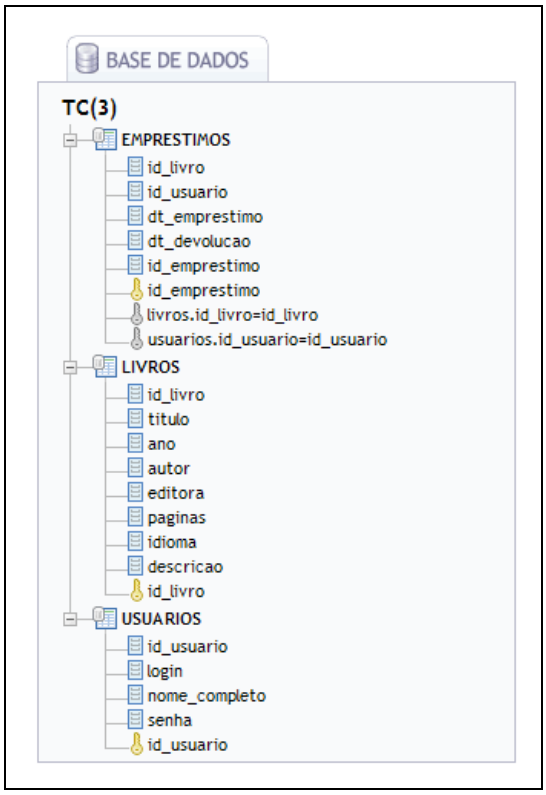


Figura 4.6 - Aba Base de Dados.

A tabela 4.1 mostra detalhadamente o significado de cada imagem representada no menu.

Tabela 4.1 Descritivo dos tipos de dados.

Imagem	Descrição
	Refere-se à tabela do banco de dados.
	Refere-se à coluna da tabela em questão.
	Refere-se à chave primária da tabela em questão.
	Refere-se às chaves estrangeiras da tabela em questão.

4.5.4 Área de edição gráfica do relatório

A área de edição gráfica do relatório é composta pelos seguintes elementos: cabeçalho, campos e rodapé.

O cabeçalho contém informações gerais do relatório como título, subtítulo, dados do autor, data de criação e número de páginas. O título e o subtítulo poderão ser editados a partir

da janela principal da ferramenta, enquanto a alteração dos demais elementos só poderá ser realizada através da janela de configurações gerais.

Para renomear o título, subtítulo do relatório ou rodapé do relatório basta clicar sobre o texto que deseja alterar. Após o clique, será apresentado um campo de formulário que possibilita a edição. Depois de aplicada a alteração, ao clicar em qualquer área da janela o processo será salvo, conforme ilustra a figura 4.7 abaixo.

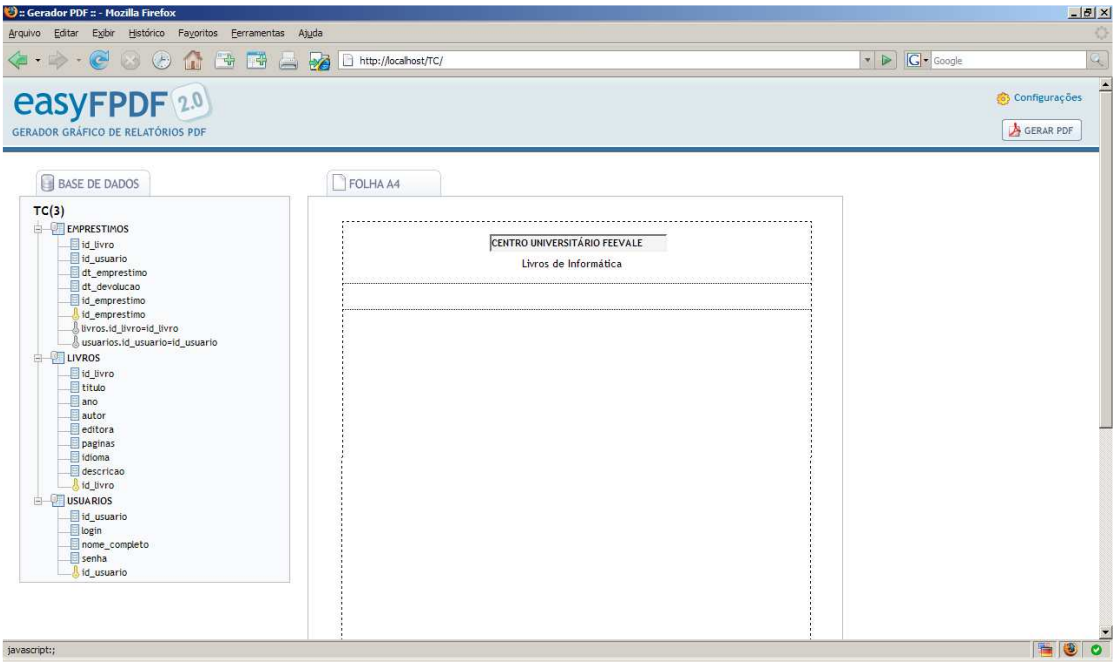


Figura 4.7 – Alterando o título do relatório.

Através da definição dos campos é que se pode relacionar os dados de uma base para o *layout* do relatório. A EasyFPDF permite que o desenvolvedor utilize o recurso *drag-and-drop*, para arrastar até a área de edição todos os campos, localizados no menu lateral Base de Dados, que deseja mostrar no relatório. A figura 4.8 apresenta a tela com um exemplo da utilização desse recurso.

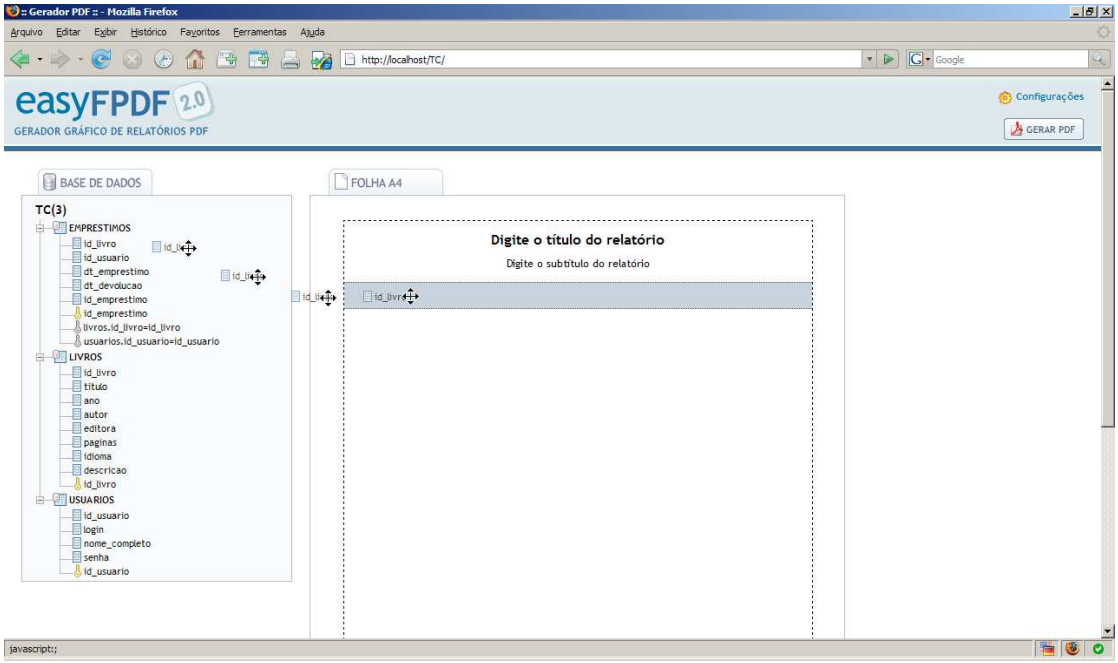


Figura 4.8 - Arrastando um campo da base de dados para a área de edição do relatório.

Outra funcionalidade muito interessante disponibilizada pela ferramenta é a opção de ordenação dos campos no relatório. O desenvolvedor poderá arrastar lateralmente os objetos, posicionando-os na área de edição do relatório conforme desejar. A figura 4.9 apresenta um exemplo.

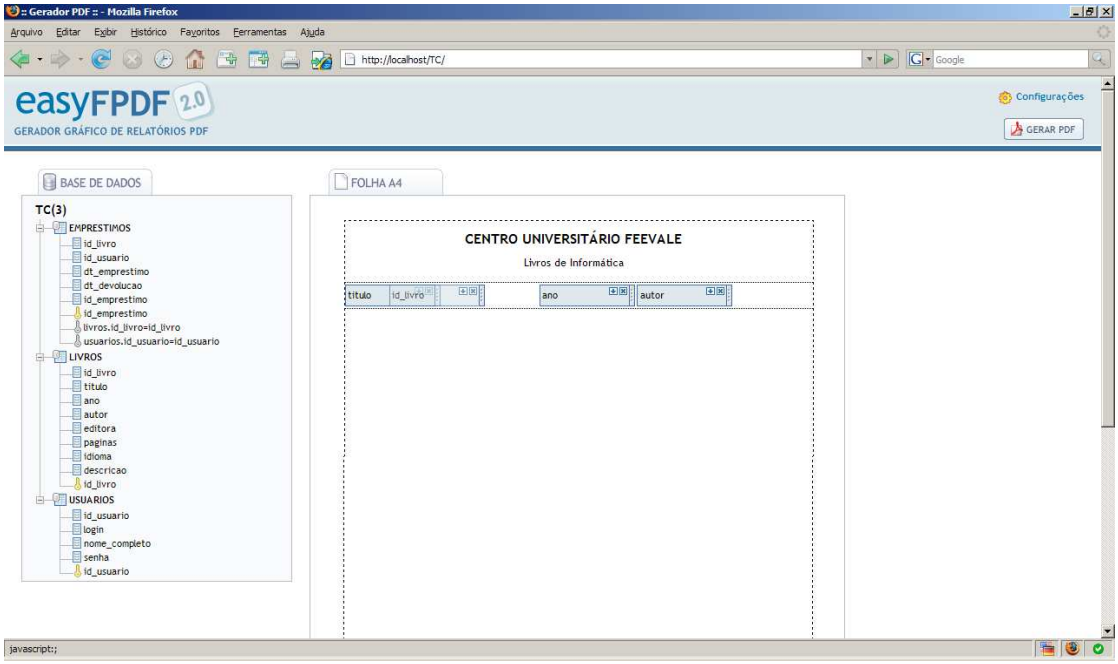


Figura 4.9 – Ordenando os campos do relatório.

4.5.5 Alterando as preferências dos campos do relatório

Assim que o relatório estiver montado, ou seja, que os campos já estiverem posicionados na área de edição gráfica, é permitido ao desenvolvedor alterar o nome e a largura do objeto em questão.

O redimensionamento do campo poderá ser feito utilizando o mouse. Basta posicioná-lo na borda direita do objeto, pressionar com o botão direito e arrastar lateralmente, conforme mostra a figura 4.10.

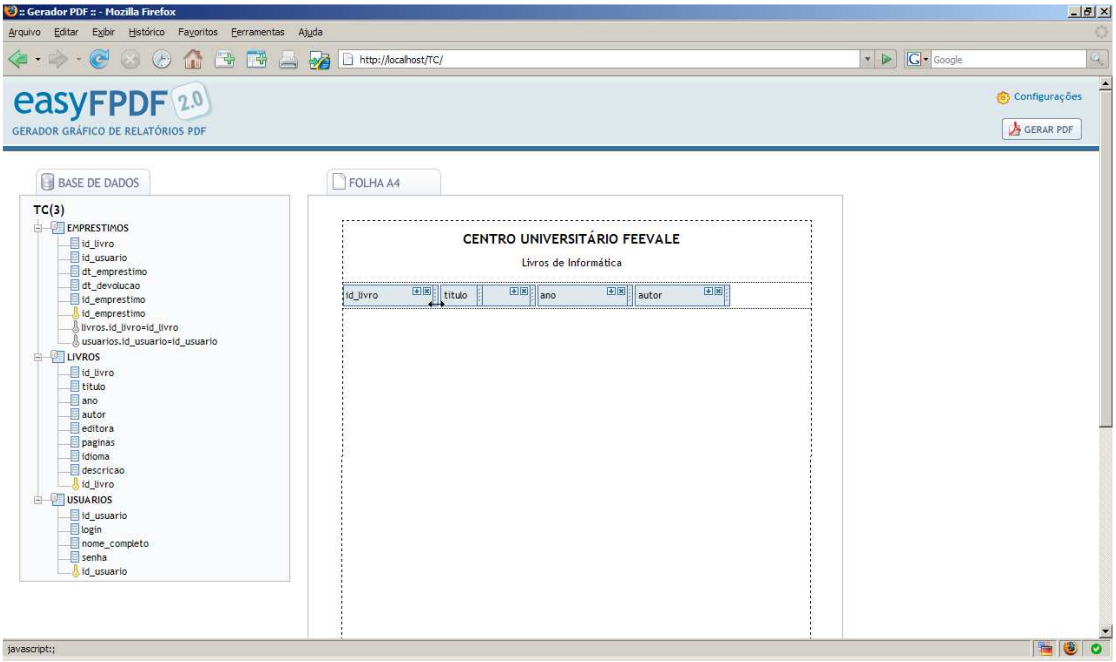


Figura 4.10 – Redimensionamento de um campo do relatório.

Para alterar o nome do campo, basta clicar na seta localizada no lado direito superior do objeto. Ao clicar, será apresentada na tela uma janela com o campo a ser alterado e um botão para salvar. A figura 4.11 ilustra um exemplo de edição do campo do relatório.

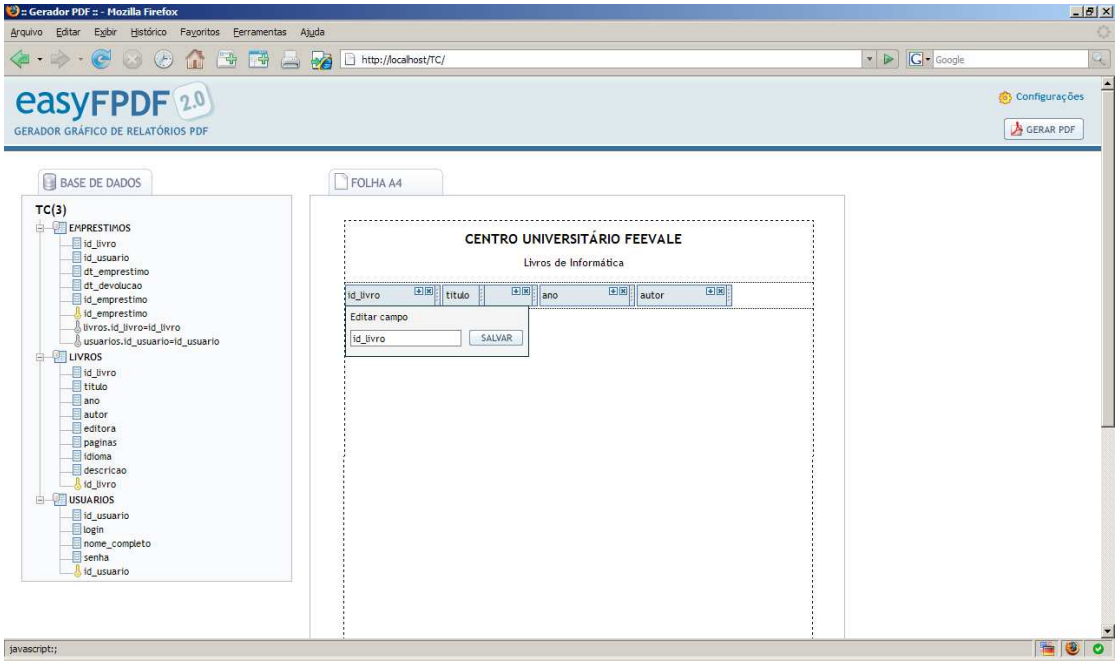


Figura 4.11 – Alterando nome do campo do relatório.

4.5.6 Gerando o PDF

Depois de realizadas todas as alterações nas preferências do relatório e de ter definido todos os campos a serem apresentados na página, é montado o documento XML que contém todas as informações de configuração do relatório. A figura 4.12 apresenta a tela com o relatório já configurado.

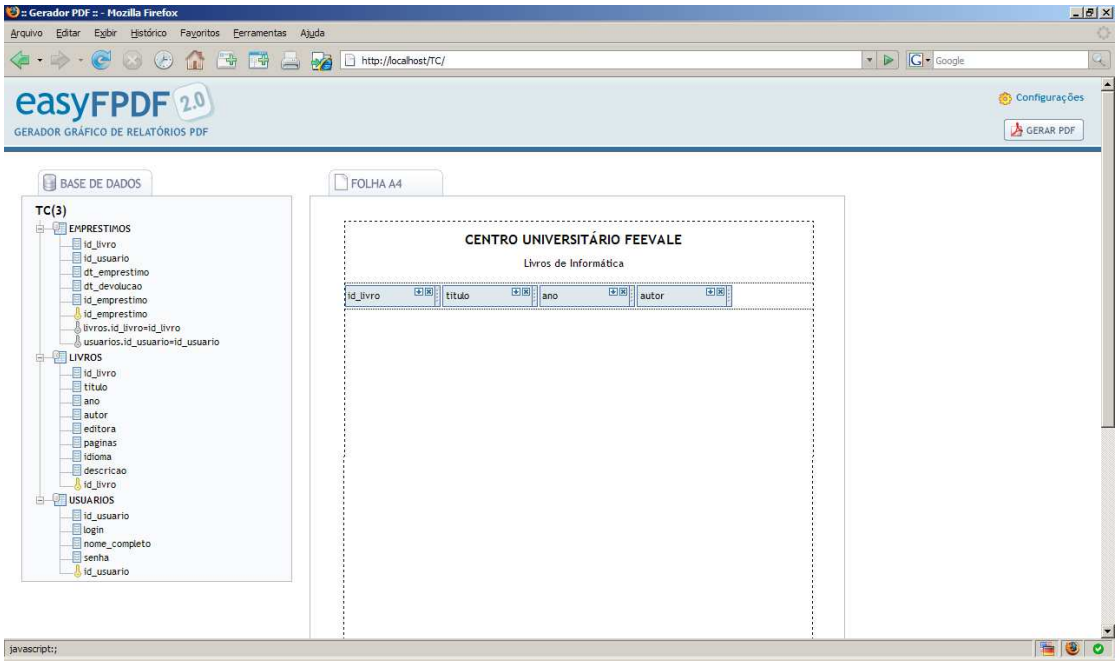


Figura 4.12 – Tela da página do relatório configurada.

Depois de criado o arquivo XML com base nas preferências definidas pelo desenvolvedor (através da interface gráfica da ferramenta), é realizada uma interação entre o documento XML e o código PHP responsável pela geração do relatório no formato PDF. O arquivo XML é apresentado no quadro 4.7 a seguir.

Quadro 4.7 – Arquivo XML de configuração do relatório.

```
1.  <?xml version="1.0" encoding="UTF-8" ?>
2.  <page>
3.    <pageConfig>
4.      <template type="simple"/>
5.      <orientation format="P"/>
6.      <margin top="2" left="2" right="2" bottom="2"/>
7.    </pageConfig>
8.    <pageHeader>
9.      <title align="center" fontColor="#000000" fontType="arial"
fontSize="12" fontBold="true" fontItalic="false">
10.       CENTRO UNIVERSITÁRIO FEEVALE
11.     </title>
12.     <subTitle align="center" fontColor="#000000" fontType="arial"
fontSize="12" fontBold="false" fontItalic="false">
13.       Livros de Informática
14.     </subTitle>
15.   </pageHeader>
16.   <pageFooter>
17.     <footer align="center" fontColor="#000000" fontType="arial"
fontSize="10" fontBold="false" fontItalic="false">
18.       Rodapé do Relatório
19.     </footer>
20.   </pageFooter>
21.   <fields>
22.     <field name="field_1" id="livros.id_livro" align="center"
width="60"> Codigo </field>
23.     <field name="field_2" id="livros.titulo" align="left" width="300">
Titulo </field>
24.     <field name="field_3" id="livros.ano" align="center" width="100">
Ano </field>
25.     <field name="field_4" id="livros.autor" align="left" width="300">
Autor </field>
26.   </fields>
27. </page>
```

A partir deste ponto, serão apresentados diversos quadros com o objetivo de explicar os processos realizados para gerar o documento PDF visualizado na figura 4.13.

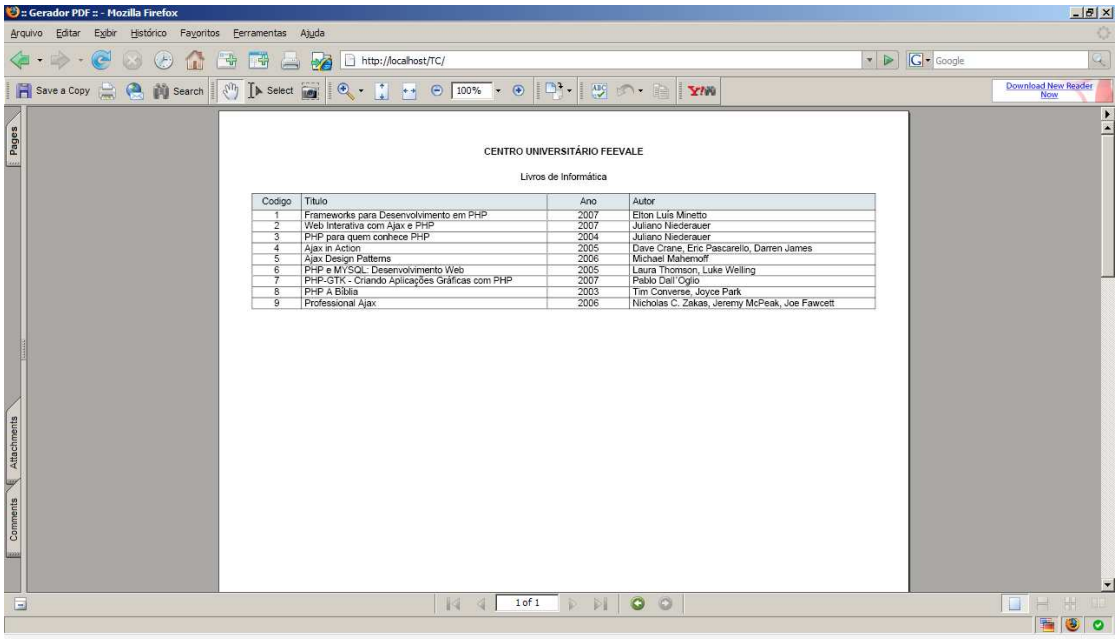


Figura 4.13 – Exemplo do relatório gerado com a EasyFPDF.

Conforme mostra o quadro 4.8 a seguir, inicialmente foi definido o diretório onde estão localizadas as fontes (linha 1) e incluídos os seguintes arquivos:

- **report.php:** Arquivo onde está definida a classe FPDF e as demais funções que foram criadas para gerar o documento PDF (linha 2);
- **kxparse.php**: Arquivo da classe KXParse, responsável pela manipulação do documento XML (linha 3);
- **adodb.inc.php**: Biblioteca ADODB, responsável pela abstração dos dados (linha 4).

Quadro 4.8 – Definição dos arquivos necessários para a geração do relatório PDF.

```
1. define('FPDF_FONTPATH','font/');
2. require_once("resources/php/fpdf/report.php");
3. require_once("resources/php/kxparse/kxparse.php");
4. require_once('resources/adodb/adodb.inc.php');
```

Após a definição dos diretórios de localização dos arquivos utilizados, é realizada a conexão com a base de dados MySQL e definido o documento XML que será consultado, conforme mostra o quadro 4.9.

Quadro 4.9 – Conexão com a biblioteca ADODB e definição do arquivo XML a ser lido.

```
1. //Criando conexão com a ADODB
2. $db= NewADOConnection("mysql");
3. $db->Connect("localhost","root","admin","TC");
4. //Definindo o arquivo XML
5. $xml=new kxparse("report.xml");
```

Realizada a conexão com a base de dados e definido o documento XML a ser lido, o próximo passo é inicializar a classe FPDF e montar a consulta SQL. O quadro 4.10 mostra esse processo.

Quadro 4.10 – Definições da consulta SQL.

```
1.  //Inicializando o FPDF
2.  $pdf=new PDF('P','mm','A4');
3.  $pdf->AddPage();
4.  //Contando o número de campos no arquivo XML
5.  $row_count=$xml->count_tag("page:fields:field","1:1:?");
6.  $row_table=$row_count + 1;
7.  //Consulta SQL
8.  $sql='select ';
9.  for ($r=1;$r<$row_table;$r++) {
10.   $sql.=$xml->get_attribute("page:fields:field","1:1:$r","id");
11.  $sql.=' as '.$xml->get_tag_text("page:fields:field","1:1:$r");
12. }
13. $sql.=' from $table';
14. //Executando o SQL
15. $result=$db->Execute($sql);
```

Conforme mostra o quadro 4.10 acima, a classe FPDF foi inicializada nas linhas 2 e 3. Em seguida utilizando o método *count_tag* (que pertence a classe KXParse) foi realizada a contagem do número de campos que serão mostrados no relatório (linhas 4 e 5). Essa contagem é necessária para montar a *query* SQL, pois baseado nela é criado um laço (entre as linhas 8 e 13) que percorre os campos e consulta as informações. Utilizam-se os métodos *get_attribute* e *get_tag_text*, para informar o nome da tabela e das colunas a serem consultadas.

Por fim, é montado um trecho de código HTML contendo a estrutura dos dados que serão mostrados no documento PDF, conforme mostra o quadro 4.11. Essa estrutura é definida na variável *$html*, que será manipulada pela função *WriteHTML*, responsável pela

montagem dos elementos na página. O código do arquivo **report.php,** onde está localizada a função *WriteHTML* pode ser visto no anexo 2.

Optou-se realizar a manipulação dessa forma, devido ao fato da configuração do relatório ser realizada através de uma interface gráfica que utiliza a linguagem HTML na definição das propriedades.

Quadro 4.11 – Montagem dos campos a serem mostrados no relatório.

```
1.  //Montando o código HTML
2.  $html='<table border="1"><tr>';
3.  for ($r=1;$r<$row_table;$r++) {
4.    $html.='<td align="'.
5.    $xml->get_attribute("page:fields:field","1:1:$r","align").'">'.
6.    $xml->get_tag_text("page:fields:field","1:1:$r").'</td>';
7.  }
8.  $html.='</tr>';
9.  while (!$result->EOF) {
10.   $pdf->SetFont('Arial','',8);
11.   $html.='<tr>';
12.   for ($r=1;$r<$row_table;$r++) {
13.     $html.='<td align="'.
14.     $xml->get_attribute("page:fields:field","1:1:$r","align").'">'.
15.     $result->fields[$r-1].
16.     '</td>';
17.   }
18.   $html.='</tr>';
19.   $result->MoveNext();
20. }
21. $pdf->WriteHTML($html);
22. $pdf->Output();
```

Neste capítulo foi apresentada uma aplicação gráfica (protótipo) para a geração de relatórios no formato PDF fazendo uso da classe FPDF. Para um melhor entendimento foram apresentados o fluxo de funcionamento, as tecnologias utilizadas no desenvolvimento, os detalhes de implementação das funcionalidades e a interface gráfica da aplicação.

CONCLUSÃO

De acordo com os estudos realizados percebeu-se que a classe FPDF oferece excelentes recursos para a geração de relatórios no formato PDF, no entanto a tarefa de desenvolvimento muitas vezes é complexa, cansativa e pouco compensadora, por necessitar que os desenvolvedores manipulem manualmente, no código PHP, a formatação, o posicionamento do texto e demais elementos que formam o relatório.

Com base em informações sobre a melhor forma de desenvolvimento de aplicações que rodam na Internet, e análise dos principais problemas encontrados na utilização da classe FPDF, partiu-se para a análise, projeto e desenvolvimento de uma ferramenta que fornecesse recursos para geração de relatórios PDF através de uma interface WEB intuitiva e, principalmente de fácil manipulação por parte dos desenvolvedores.

De posse desse conhecimento, foi realizado um levantamento das tecnologias WEB que melhor poderiam se enquadrar nesse estudo. Além disso, foi apresentada a mudança de comportamento no desenvolvimento de aplicações WEB, onde a utilização de técnicas, como o AJAX, propõe uma experiência de uso semelhante à de aplicativos *desktop*.

Durante o desenvolvimento da ferramenta foram encontradas algumas dificuldades na utilização do AJAX, principalmente no que diz respeito à compatibilidade das suas funcionalidades entre os navegadores testados: Internet Explorer 6, Internet Explorer 7, Mozilla Firefox e Opera. Essas dificuldades estão relacionadas ao fato de não existir um padrão universal de interpretação das linguagens Javascript e CSS por parte dos navegadores. No entanto, ao seguir as recomendações especificadas pela W3C, muitos dos problemas enfrentados foram resolvidos, fazendo com que a aplicação ficasse 100% operacional nos navegadores testados, sendo necessário alguns ajustes a nível de *layout* que podem ser realizados em um trabalho futuro.

Algumas limitações na interação entre o arquivo XML e a biblioteca ADODB foram encontradas, por exemplo: junções entre tabelas na consulta SQL podem ser realizadas utilizando uma classe chamada *Oh!DB*. Para utilizar essa classe é necessário implementar um módulo específico para que seja feita a interação com a ADODB. Por não haver tempo hábil para essa implementação, fica como sugestão para um trabalho futuro.

Por fim, propõe-se uma série de desenvolvimentos e implementações futuras para aperfeiçoamento e seqüência do trabalho aqui descrito. Entre estas, citam-se:

1. Implementar um módulo de configuração da base de dados a ser utilizada na aplicação;

2. Possibilitar ao desenvolvedor salvar, alterar e excluir os relatórios criados;

3. Incluir novos *templates* para a criação de relatórios;

4. Incluir uma opção para visualização das consultas SQL a serem realizadas;

5. Possibilitar a definição da ordenação dos dados das colunas das tabelas;

6. Possibilitar a seleção de funções (contagem, soma e média) e agrupamento de dados;

7. Incluir uma opção para a definição de parâmetros a serem utilizados nas consultas SQL;

8. Possibilitar a definição de máscaras nos dados a serem exibidos, como por exemplo, valores monetários.

Conclui-se, portanto que a EasyFPDF é uma ferramenta viável para a geração de relatórios PDF, e útil para um aumento significativo de produtividade na implementação dos mesmos por parte dos desenvolvedores que utilizam a classe FPDF.

REFERÊNCIAS BIBLIOGRÁFICAS

ADOBE. **Adobe Reader**, 2007a. Disponível em: <http://www.adobe.com/br/products/acrobat/readstep2.html>. Acesso em: 02 jun. 2007.

ADOBE. **O que é Adobe PDF?**, 2007b. Disponível em: <http://www.adobe.com/br/products/acrobat/messaging/pdf.html>. Acesso em: 21 abr. 2007.

ADOBE. **PDF Reference**. 6 ed. Adobe Systems Incorporated, 2006. 1289p.

ADODB. **ADOdb Database Abstraction Library for PHP**, 2007. Disponível em: <http://adodb.sourceforge.net>. Acesso em: 10 set. 2007.

AJAXLIB. **AJAXLib**, 2007. Disponível em: <http://sourceforge.net/projects/ajaxlib>. Acesso em: 21 mai. 2007.

ALENCAR, Paulo de. Na Aché o PDF ajuda a fazer remédios. **Coleção INFO**, São Paulo, ed. 30, p. 20 – 22, jun. 2006.

ASPPDF. **What is AspPDF?**, 2007. Disponível em: <http://www.fpdf.org/>. Acesso em: 1 mai. 2007.

BASIURA, Russ et al., **Professional ASP .NET Web Service**. São Paulo: Pearson Education, 2003. 725p.

BIRT. **Business Intelligence and Reporting Tools**, 2007. Disponível em: <http://www.eclipse.org/birt/phoenix/> Acesso em: 26 abr. 2007.

BUSINESS OBJECTS. **Crystal Reports**, 2007. Disponível em: <http://www.businessobjects.com/products/reporting/crystalreports/default.asp> Acesso em 14 abr. 2007.

CARDOSO, André. O banco fala sqlquês. **Coleção INFO**, São Paulo, ed. 27, p. 31 – 33, mar. 2006.

CARTER, Sandy. **New Language of Business, The**: SOA & Web 2.0. IBM Press, 2007. 320p.

CAVALCANTI, Marcos; NEPOMUCENO, Carlos. **O conhecimento em rede**. São Paulo: Campus, 2007. 134p.

CHOI, Wankyu et al. **Beginning PHP4 Programando**. São Paulo: Makron Books, 2001. 719p.

CIPRIANI, Fabio. **Blog Corporativo**. São Paulo: Novatec, 2006. 208p.

CONVERSE, Tim; PARK, Joyce. **PHP 4 a Bíblia**. São Paulo: Campus, 2001. 697p.

CONVERSE, Tim; PARK, Joyce; MORGAN, Clark. **PHP5 and MySQL Bible**. Wiley, 2006. 1083p.

CRANE, Dave; PASCARELLO, Eric; JAMES, Darren. **Ajax In Action**. Manning, 2006. 650p.

DALL'OGLIO, Pablo. PHP-GTK Aplicações Gráficas em PHP. **Geek**, São Paulo, ed. 38, p.26 – 29. dez. 2003.

DARIE, Cristian et al. **Building Responsive Web Applications**. Packt, 2006. 273p.

DAUM, Berthold; UDO, Merten. **Arquitetura de Sistemas com XML**. São Paulo: Campus, 2002. 464p.

DAVENPORT, Thomas H.; MARCHAND, Donald A.; DICKSON, Tim. **Dominando a gestão da informação**. Porto Alegre: Bookman, 2004. 407p.

DEITEL, H. M.; DEITEL, P. J.; NIETO, T. R. **Intenet e World Wide Web Como Programar**. ed. 2. Porto Alegre: Bookman, 2003. 1274p.

DEVMEDIA. **DevMedia Cursos Online**. Disponível em: <http://www.devmedia.com.br/cursos/studentportal.asp?curso=3&topico=33&ord=2>. Acesso em: 5 mai. 2007.

DOJO. **Dojo**, 2007. Disponível em: <http://dojotoolkit.org/>. Acesso em: 27 mai. 2007.

DWR. **Direct Web Remoting**, 2007. Disponível em: <http://getahead.org/dwr>. Acesso em: 22 mai. 2007.

FLANAGAN, David. **JavaScript**: The Definitive Guide. O'Reilly, 2004. 818p.

FLICKR. **Flickr**, 2007. Disponível em: <http://www.flickr.com/>. Acesso em: 4 jun. 2007.

FPDF. **FPDF**, 2007. Disponível em: <http://www.fpdf.org/>. Acesso em: 10 mar. 2007.

FUECKS, Harry. **The PHP Anthology: Object Oriented PHP Solutions, Vol.2 –** Applications. SitePoint, 2003. 438p.

FUNG, Khun Yee**. Working with XML and HTML**. Addison-Wesley, 2001. 441p.

GARRET J. J. **AJAX: A New Approach to Web Applications**, 2005. Disponivel em: <http://www.adaptivepath.com/publications/essays/archives/000385.php>. Acesso em 24 mai. 2007.

GREENSPAN, Jay; BULGER, Brad. **MySQL/PHP Database Applications**. M&T Books, 2001. 586p.

GREGO, Mauricio. Do PostScript ao 3D. **Coleção INFO**, São Paulo, ed. 30, p. 7 – 12, jun. 2006.

GREGORIO, Danilo. Economia na Seguradora. **Coleção INFO**, São Paulo, ed. 30, p. 17 – 19, jun. 2006a.

GREGORIO, Danilo. O Estadão viaja na Web. **Coleção INFO**, São Paulo, ed. 30, p. 23 –25, jun. 2006b.

GWT. **Google Web Toolkit**, 2007. Disponível em: <http://code.google.com/webtoolkit/>. Acesso em: 21 mai. 2007.

HAMMERSLEY, B., **Content Syndication with RSS**. O'Reilly, 2003. 222p.

KXPARSE. **KXParse**, 2007. Disponível em: < http://kxparse.berlios.de>. Acesso em: 2 set. 2007.

LAUDON, Kenneth C.; LAUDON, Jane P. **Sistemas de informação: com internet**. 4.ed. Rio de Janeiro: LTC, 2001. 389p.

LEME, Felipe. JSTL – Guia Completo. **Java Magazine**, Rio de Janeiro, ed. 7, p. 34 – 40, jun. 2003.

LERDOF, Rasmus; TRATOE, Kevin. **Programming PHP**. O'Reilly, 2002. 508p.

LOZANO, Fernando. Relatórios Corporativos com JasperReports e iReport. **Java Magazine**, Rio de Janeiro, ed. 13, p. 20 – 32, abr. 2005.

MAHEMOFF, Michael. **Ajax Design Patterns**. O'Reilly, 2006. 655p.

MINETTO, Elton L. **Frameworks para Desenvolvimento em PHP**. São Paulo: Novatec, 2007. 192 p.

MONTOYA, Carlos E. **Crystal Reports 10**: Guia de Procedimentos Avançados. Santa Catarina: Visual Books, 2005. 285p.

MYSQL. **MySQL**, 2007. Disponível em: <http://www.mysql.com>. Acesso em: 25 out. 2007.

MUSSER, John. **Web 2.0 Principles and Best Practices**. O'Reilly, 2006. 101p.

NIEDERAUER, Juliano. **PHP para quem conhece PHP**. São Paulo: Novatec, 2004. 480 p.

NIEDERAUER, Juliano. **Web Interativa com Ajax e PHP**. São Paulo: Novatec, 2007. 288p.

NIEDERAUER, Juliano; PRATES, Rubens. **MySQL 5**: Guia de Consulta Rápida. São Paulo: Novatec, 2006. 112p.

O'BRIEN, James A. **Sistemas de informação e as decisões gerenciais na era da internet**. 2. ed. São Paulo: Saraiva, 2006. 431p.

OLIVEIRA, Wilson J. de. **ColdFusion**: Desenvolvendo Sites Profissionais. Santa Catarina: Visual Books, 2002. 194p.

O'REILLY, Tim. **What is Web 2.0?**. 2005. Disponível em: <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>. Acesso em: 24 mar. 2007.

PDFLIB. **PDFlib**, 2007. Disponível em: <http://www.pdflib.com/>. Acesso em: 1 mai. 2007.

PHP. **Manual do PHP**, 2007. Disponível em: <http://www.php.net/manual>. Acesso em: 31 mar. 2007.

PORTO SEGURO. **Porto Seguro Seguros**, 2007. Disponível em: <http://www.portoseguro.com.br/site/institucional/historia.cfm>. Acesso em: 12 mai. 2007.

PROTOTYPE. **Prototype Javascript Framework**, 2007. Disponível em: <http://www.prototypejs.org/>. Acesso em: 27 mai. 2007.

REZENDE, Denis A.; ABREU, Aline F. de. **Tecnologia da informação aplicada a sistemas de informação empresariais**: o papel estratégico da informação e dos sistemas de informação nas empresas. São Paulo: Atlas, 2000. 311 p.

RUAS, Nilson. **Criando Sites Web com HTML 4**. Santa Catarina: Visual Books, 2002. 88 p.

SAADE, Joel. **Crystal Reports**: Guia de Consulta Rápida. São Paulo: Novatec, 2001. 127 p.

SAJAX. **SAJAX**, 2007. Disponível em: <http://www.modernmethod.com/sajax/>. Acesso em: 21 mai. 2007.

SCRIPTACULOUS. **Script.aculo.us**, 2007. Disponível em: <http://script.aculo.us>. Acesso em: 21 mai. 2007.

SILVA FILHO, Antonio M. **Programando com XML**. Rio de Janeiro: Elsevier, 2004. 332 p.

SUN. **JavaServer Pages Technology**, 2007. Disponível em: <http://java.sun.com/products/jsp/>. Acesso em 1 mai. 2007.

SYMFONY. **Symfony Project**, 2007. Disponível em: <http://www.symfony-project.com/>. Acesso em: 21 mai. 2007.

VILLAS, Marcos Viana.; VILLASBOAS Luiz Felipe P. **Programação Técnicas e Linguagens**. Rio de Janeiro: Campus. 1988. 195p.

ZAKAS, Nicholas C.; MCPEAK, Jeremy; FAWCET, Joe. **Professional Ajax**. Wiley, 2006. 432p.

XAJAX. **XAJAX**, 2007. Disponível em: <http://www.xajaxproject.org/>. Acesso em: 24 mai. 2007.

W3CSCHOOLS. **Browser Statistics**, 2007. Disponível em: <http://www.w3schools.com/browsers/browsers_stats.asp >. Acesso em: 2. nov. 2007.

WCAG. **Web Content Accessibility Guidelines 1.0**, 2007. Disponível em: <http://www.w3.org/TR/WAI-WEBCONTENT>. Acesso em: 20 out. 2007.

WEBSUPERGOO. **ABCpdf**, 2007. Disponível em: <http://www.websupergoo.com/abcpdf-1.htm>. Acesso em: 3 mai. 2007.

WELLING, Luke; THOMSON, Laura. **PHP e MySQL**: desenvolvimento WEB. Rio de Janeiro: Campus, 2003. 676 p.

ANEXO 1 – EXEMPLO DA GERAÇÃO DE UM RELATÓRIO PDF UTILIZANDO A CLASSE FPDF

```
<?php
define('FPDF_FONTPATH','font/');
require('fpdf.php');

class PDF extends FPDF {
       function Header() {
            $this->SetFont('Arial','B',12);
            $this->SetY(10);
            $this->Cell(23);
            $this->Cell(160,10,'GERADOR GRÁFICO DE RELATÓRIOS PDF - CATÁLOGO
DE LIVROS',1,0,'C');
       }
       function Footer() {
            $this->SetY(-15);
            $this->SetFont('Arial','',8);
            $this->Cell(0,10,'Página '.$this->PageNo(),0,0,'C');
       }
}

/*******************************************************
       Conexão e consulta no Mycatalogo
*******************************************************/
include ("mysql_connect.php");
include ("mysql_execute.php");
$catalogo = "select * from livros order by titulo";
$resultado = mysqlexecute($id,$catalogo);
//Registros por página
$registros_pagina = 7;

//Criando novo PDF
$pdf=new PDF('P','mm','A4');
//Definindo margens
$pdf->SetMargins(2,2,2);
//Inicializando arquivo
$pdf->Open();
//Habilitando quebra automática de página
$pdf->SetAutoPageBreak(false);
//Inserindo primeira página
$pdf->AddPage();
//Definindo autor
$pdf->SetAuthor('Igor Henrique Berlitz');
//Definindo título do documento
$pdf->SetTitle('Catálogo de Livros');
```

```
/*******************************************************
       Definindo parâmetros de posicionamento na página
*******************************************************/
//Tamanho da célula da imagem
$altura_celula_img = 30;
$largura_celula_img = 31;
//Posicionamento da célula da imagem
$pos_horizontal_celula_img = 25;
$pos_vertical_celula_img = 25;
//Posicionamento da imagem
$posicao_horizontal_img = $pos_horizontal_celula_img + 1;
$posicao_vertical_img = $pos_vertical_celula_img + 5;
//Posicionamento do titulo
$pos_horizontal_titulo = 25;
$posicao_vertical_titulo = 55;
//Posicionamento da célula dos dados
$pos_horizontal_info = 55;
$pos_vertical_info = 32;
//Posicionamento do texto
$pos_horizontal_autor = 37;
$pos_horizontal_ano = 41;
$pos_horizontal_editora = 45;
$pos_horizontal_paginas = 49;
$pos_horizontal_idioma  = 53;
$pos_vertical_texto = 57;
//Espaçamento entre linhas
$espaco_linha = 35;

//Inicializando contador
$i = 0;

while($row = mysql_fetch_array($resultado)) {

       if ($i == $registros_pagina) {
            $pdf->AddPage();
            //Posicionamento da célula da imagem
            $pos_horizontal_celula_img = 25;
            $pos_vertical_celula_img = 25;
            //Posicionamento da imagem
            $posicao_horizontal_img = $pos_horizontal_celula_img + 1;
            $posicao_vertical_img = $pos_vertical_celula_img + 5;
            //Posicionamento do titulo
            $pos_horizontal_titulo = 25;
            $posicao_vertical_titulo = 55;
            //Posicionamento da célula dos dados
            $pos_vertical_info = 32;
            $pos_horizontal_info = 55;
            //Posicionamento do texto
            $pos_vertical_texto= 57;
            $pos_horizontal_autor    = 37;
            $pos_horizontal_ano= 41;
            $pos_horizontal_editora = 45;
            $pos_horizontal_paginas= 49;
            $pos_horizontal_idioma  = 53;
            $i = 0;
       }

       //Campos da consulta
       $titulo  = $row['titulo'];
       $autor   = 'Autor: '.$row['autor'];
       $imagem  = 'images/'.$row['codigo'].'.jpg';
```

```
        $ano     = 'Ano de Publicação: '.$row['ano'];
        $editora = 'Editora: '.$row['editora'];
        $paginas = 'Páginas: '.$row['paginas'];
        $idioma  = 'Idioma: '.$row['idioma'];

        //Célula da imagem
        $pdf->SetY($pos_horizontal_celula_img);
        $pdf->SetX($pos_vertical_celula_img);

        $pdf->SetFont('Arial','',9);
        $pdf->Cell($altura_celula_img,$largura_celula_img,'',1,'','0');

        //Inserindo imagem
        $pdf-
>Image($imagem,$posicao_vertical_img,$posicao_horizontal_img,'20' ,''
,'JPG','');

        //Célula do titulo
        $pdf->SetY($pos_horizontal_titulo);
        $pdf->SetX($posicao_vertical_titulo);
        $pdf->SetFont('Arial','',9);
        $pdf->Cell(130,7,$titulo,1,1,'L');

        //Célula dos dados
        $pdf->SetY($pos_vertical_info);
        $pdf->SetX($pos_horizontal_info);
        $pdf->SetFont('Arial','',9);
        $pdf->Cell(130,24,'',1,1,'C');

        //Inserindo dados
        $pdf->SetFont('Arial','',7);
        $pdf->Text($pos_vertical_texto,$pos_horizontal_autor,$autor);
        $pdf->Text($pos_vertical_texto,$pos_horizontal_ano,$ano);
        $pdf->Text($pos_vertical_texto,$pos_horizontal_editora,$editora);
        $pdf->Text($pos_vertical_texto,$pos_horizontal_paginas,$paginas);
        $pdf->Text($pos_vertical_texto,$pos_horizontal_idioma,$idioma);

/*************************************************************************
        Incrementando valores dos  parâmetros para posicionamento na página
*************************************************************************/
        $pos_horizontal_celula_img = $pos_horizontal_celula_img +
$espaco_linha;
        $posicao_horizontal_img    = $posicao_horizontal_img +
$espaco_linha;
        $pos_horizontal_titulo     = $pos_horizontal_titulo + $espaco_linha;
        $pos_vertical_info         = $pos_vertical_info + $espaco_linha;
        $pos_horizontal_autor      = $pos_horizontal_autor + $espaco_linha;
        $pos_horizontal_ano        = $pos_horizontal_ano+ $espaco_linha;
        $pos_horizontal_editora    = $pos_horizontal_editora +
$espaco_linha;
        $pos_horizontal_paginas    = $pos_horizontal_paginas+ $espaco_linha;
        $pos_horizontal_idioma     = $pos_horizontal_idioma + $espaco_linha;
        $i = $i + 1;
}

//Criando arquivo
$pdf->Output();
?>
```

ANEXO 2 – CÓDIGO DA PÁGINA REPORT.PHP

```
<?php

require('fpdf.php');

function hex2dec($color = "#000000"){
	$R = substr($color, 1, 2);
	$red = hexdec($R);
	$V = substr($color, 3, 2);
	$green = hexdec($V);
	$B = substr($color, 5, 2);
	$blue = hexdec($B);
	$tbl_color = array();
	$tbl_color['R']=$red;
	$tbl_color['G']=$green;
	$tbl_color['B']=$blue;
	return $tbl_color;
}

//converte pixel -> milimitro = 72 dpi
function px2mm($px){
	return $px*25.4/72;
}

function txtentities($html){
	$trans = get_html_translation_table(HTML_ENTITIES);
	$trans = array_flip($trans);
	return strtr($html, $trans);
}

class PDF extends FPDF
{
var $B;
var $I;
var $U;
var $HREF;
var $fontList;
var $issetfont;
var $issetcolor;

function PDF($orientation='P',$unit='mm',$format='A4')
{
	$this->FPDF($orientation,$unit,$format);
	$this->B=0;
	$this->I=0;
	$this->U=0;
	$this->HREF='';
```

```
        $this->tableborder=0;
        $this->tdbegin=false;
        $this->tdwidth=0;
        $this->tdheight=0;
        $this->tdalign="L";
        $this->tdbgcolor=false;
        $this->oldx=0;
        $this->oldy=0;
        $this-
>fontlist=array("arial","times","courier","helvetica","symbol");
        $this->issetfont=false;
        $this->issetcolor=false;
}

//HTML Parser
function WriteHTML($html)
{
        //remove tags incompativeis
        $html=strip_tags($html,"<b><u><i><a><img><p><br><strong><em>
        <font><tr><blockquote><hr><td><tr><table><sup>");
        $html=str_replace("\n",'',$html);
        $html=str_replace("\t",'',$html);
        //separa string
        $a=preg_split('/<(.*)>/U',$html,-1,PREG_SPLIT_DELIM_CAPTURE);
        foreach($a as $i=>$e)
        {
             if($i%2==0)
             {
                   //Texto
                   if($this->HREF)
                         $this->PutLink($this->HREF,$e);
                   elseif($this->tdbegin) {
                         if(trim($e)!='' and $e!=" ") {
                               $this->Cell($this->tdwidth,$this-
>tdheight,$e,$this->tableborder,'',
                               $this->tdalign,$this->tdbgcolor);
                         }
                         elseif($e==" ") {
                               $this->Cell($this->tdwidth,$this-
>tdheight,'',$this->tableborder,'',
                               $this->tdalign,$this->tdbgcolor);
                         }
                   }
                   else
                         $this->Write(5,stripslashes(txtentities($e)));
             }
             else
             {
                   //Tags
                   if($e{0}=='/')
                         $this->CloseTag(strtoupper(substr($e,1)));
                   else
                   {
                         //Extrai atributos
                         $a2=explode(' ',$e);
                         $tag=strtoupper(array_shift($a2));
                         $attr=array();
                         foreach($a2 as $v)

        if(ereg('^([^=]*)=["\']?([^"\']*)["\']?$',$v,$a3))
```

```
                                    $attr[strtoupper($a3[1])]=$a3[2];
                        $this->OpenTag($tag,$attr);
                  }
            }
      }
}

function OpenTag($tag,$attr)
{
      //Abre tag
      switch($tag){

            case 'SUP':
                  if($attr['SUP'] != '') {
                        $this->Cell(2,2,$attr['SUP'],0,0,'L');
                  }
                  break;

            case 'TABLE':
                  if( $attr['BORDER'] != '' ) $this-
>tableborder=$attr['BORDER'];
                  else $this->tableborder=0;
                  break;
            case 'TR':
                  break;
            case 'TD':
                  if( $attr['WIDTH'] != '' ) $this-
>tdwidth=($attr['WIDTH']/4);
                  else $this->tdwidth=40;
                  if( $attr['HEIGHT'] != '') $this-
>tdheight=($attr['HEIGHT']/6);
                  else $this->tdheight=6;
                  if( $attr['ALIGN'] != '' ) {
                        $align=$attr['ALIGN'];
                        if($align=="left") $this->tdalign="L";
                        if($align=="center") $this->tdalign="C";
                        if($align=="right") $this->tdalign="R";
                  }
                  else $this->tdalign="L";
                  if( $attr['BGCOLOR'] != '' ) {
                        $coul=hex2dec($attr['BGCOLOR']);
                              $this-
>SetFillColor($coul['R'],$coul['G'],$coul['B']);
                              $this->tdbgcolor=true;
                        }
                  $this->tdbegin=true;
                  break;

            case 'HR':
                  if( $attr['WIDTH'] != '' )
                        $Width = $attr['WIDTH'];
                  else
                        $Width = $this->w - $this->lMargin-$this->rMargin;
                  $x = $this->GetX();
                  $y = $this->GetY();
                  $this->SetLineWidth(0.2);
                  $this->Line($x,$y,$x+$Width,$y);
                  $this->SetLineWidth(0.2);
                  $this->Ln(1);
                  break;
            case 'STRONG':
```

```
                  $this->SetStyle('B',true);
                  break;
            case 'EM':
                  $this->SetStyle('I',true);
                  break;
            case 'B':
            case 'I':
            case 'U':
                  $this->SetStyle($tag,true);
                  break;
            case 'A':
                  $this->HREF=$attr['HREF'];
                  break;
            case 'IMG':
                  if(isset($attr['SRC']) and (isset($attr['WIDTH']) or
isset($attr['HEIGHT']))) {
                        if(!isset($attr['WIDTH']))
                              $attr['WIDTH'] = 0;
                        if(!isset($attr['HEIGHT']))
                              $attr['HEIGHT'] = 0;
                        $this->Image($attr['SRC'], $this->GetX(), $this-
>GetY(), px2mm($attr['WIDTH']),
                        px2mm($attr['HEIGHT']));
                  }
                  break;
            //case 'TR':
            case 'BLOCKQUOTE':
            case 'BR':
                  $this->Ln(5);
                  break;
            case 'P':
                  $this->Ln(10);
                  break;
            case 'FONT':
                  if (isset($attr['COLOR']) and $attr['COLOR']!='') {
                        $coul=hex2dec($attr['COLOR']);
                        $this-
>SetTextColor($coul['R'],$coul['G'],$coul['B']);
                        $this->issetcolor=true;
                  }
                  if (isset($attr['FACE']) and
in_array(strtolower($attr['FACE']), $this->fontlist)) {
                        $this->SetFont(strtolower($attr['FACE']));
                        $this->issetfont=true;
                  }
                  if (isset($attr['FACE']) and
in_array(strtolower($attr['FACE']), $this->fontlist)
                  and isset($attr['SIZE']) and $attr['SIZE']!='') {
                        $this-
>SetFont(strtolower($attr['FACE']),'',$attr['SIZE']);
                        $this->issetfont=true;
                  }
                  break;
      }
}

function CloseTag($tag)
{
      //Fecha tag
      if($tag=='SUP') {
      }
```

```
        if($tag=='TD') {
             $this->tdbegin=false;
             $this->tdwidth=0;
             $this->tdheight=0;
             $this->tdalign="L";
             $this->tdbgcolor=false;
        }
        if($tag=='TR') {
             $this->Ln();
        }
        if($tag=='TABLE') {
             $this->tableborder=0;
        }

        if($tag=='STRONG')
             $tag='B';
        if($tag=='EM')
             $tag='I';
        if($tag=='B' or $tag=='I' or $tag=='U')
             $this->SetStyle($tag,false);
        if($tag=='A')
             $this->HREF='';
        if($tag=='FONT'){
             if ($this->issetcolor==true) {
                  $this->SetTextColor(0);
             }
             if ($this->issetfont) {
                  $this->SetFont('arial');
                  $this->issetfont=false;
             }
        }
}

function SetStyle($tag,$enable)
{
        $this->$tag+=($enable ? 1 : -1);
        $style='';
        foreach(array('B','I','U') as $s)
             if($this->$s>0)
                  $style.=$s;
        $this->SetFont('',$style);
}

function PutLink($URL,$txt)
{
        $this->SetTextColor(0,0,255);
        $this->SetStyle('U',true);
        $this->Write(5,$txt,$URL);
        $this->SetStyle('U',false);
        $this->SetTextColor(0);
}

}

?>
```