**PATRICIA DIAS**

# UTILIZANDO XML PARA A TROCA DE INFORMAÇÕES ENTRE EMPRESAS

**JOINVILLE**

**2004**

**PATRICIA DIAS**

# UTILIZANDO XML PARA A TROCA DE INFORMAÇÕES ENTRE EMPRESAS

Trabalho de Conclusão de Curso submetido ao Instituto Superior Tupy como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação, sob a orientação do professor Marcos Laureano, MSc.

Joinville

2004

**UTILIZANDO XML PARA A TROCA DE INFORMAÇÕES ENTRE EMPRESAS**

Patrícia Dias

Este trabalho de conclusão de curso foi julgado adequado para obtenção do título de Bacharel em Sistemas de Informação, e aprovada em sua forma final pelo Instituto Superior Tupy.

Joinville, ____ de dezembro de 2004.

_____

Marcos Laureano, MSc.

_____

Marco André Lopes Mendes, MSc.

Banca Examinadora:

_____

Zelindo Petri, Especialista

_____

Maristela Regina Weinfurter, MSc.

## AGRADECIMENTOS

A Deus, que por sua imensa bondade, torna tudo possível.

A minha mãe, Sueli, pelo incentivo.

Ao meu noivo, Maico, pelo carinho, paciência e compreensão.

A minha amiga, Gisele, por esses quatro anos de convivência e, em especial, pelo incentivo nesta fase final.

Aos demais amigos e familiares que, de forma direta ou indireta, ajudaram-me nesta conquista.

## RESUMO

A *eXtensible Markup Language* (XML) é uma tecnologia capaz de executar com excelência o intercâmbio de informações entre parceiros comerciais. Durante vários anos a tecnologia disponível para a realização desta comunicação era o EDI (*Eletronic Data Interchange*), cujas limitações vem abrindo caminho para a crescente adoção da XML pelas empresas. Com isso, este trabalho visa apresentar os benefícios oferecidos por esta recente tecnologia, assim como um estudo de caso que demonstre a sua utilização.

**ABSTRACT**

The Extensible Markup Language(XML) is a technology capable to execute with excellence the information interchange between the commercial partners. During several years the only available technology able to make this communication was EDI (Eletronic Data Interchange), this limitation is opening way to the growing adapt ion of the XML by the companies. According to this research has the porpoise to show the benefits offered by the new technology through the study of this case, that present its utility.

**SUMÁRIO**

## LISTA DE ABREVIATURAS

| | |
|---|---|
| DOM | Document Object Model |
| DTD | Document Type Definition |
| EDI | Eletronic Data Interchange |
| HTML | HiperText Transfer Protocol |
| SGML | Standard Generalized Markup Language |
| SQL | Structured Query Language |
| URL | Uniform Resource Locator |
| W3C | Word Wide Web Consortium |
| XML | Extensible Markup Language |
| XSL | Extensible Stylesheet Language |
| SSL | Secure Sockets Layer |

# LISTA DE FIGURAS

# INTRODUÇÃO

O presente estudo foi desenvolvido de forma a detalhar os benefícios oferecidos pela XML (*Extensible Markup Language)* e descrever a eficiência desta recente tecnologia na troca de informações entre aplicações que utilizam diferentes bancos de dados. A utilização de documentos XML em substituição aos arquivos gerados por meio do processo de EDI tende a representar ganhos significativos para empresas que pretendem criar ou aperfeiçoar a comunicação eletrônica com seus parceiros comerciais.

A XML surgiu em 1996 de forma a suprir a necessidade de uma linguagem de marcação que fosse fácil de ser implementada na Internet e passível de expansão e personalização. A partir de então, as vantagens oferecidas por esta recente tecnologia como simplicidade, flexibilidade, extensibilidade e interoperabilidade tornaram crescente o número de usuários e aplicações que passaram a adota-la como forma de manipulação, representação e transporte de dados.

Até então, a principal forma de trocar informações comerciais entre organizações era o EDI. Porém, além de possuir um alto custo, esta solução incorpora rígidas regras comerciais e necessita de um novo trabalho de engenharia a cada novo parceiro comercial que a organização deseja agregar.

Constituem-se os objetivos desta pesquisa: conhecer melhor a tecnologia XML; descrever como a troca de informações entre empresas pode ser simplificada com o uso da mesma e exemplificar a integração de documentos XML com um banco de dados.

A metodologia utilizada para o desenvolvimento deste estudo foi bibliográfica, sendo as principais fontes de consulta livros, artigos e Internet. Para a realização do estudo de caso foram aplicados os conceitos previamente descritos.

Este estudo está dividido em 4 capítulos. O capítulo 1 apresenta a definição de XML, um breve histórico e expõe alguns conceitos básicos sobre esta tecnologia, incluindo suas vantagens e desvantagens. O capítulo 2 abrange o EDI, sua definição, suas vantagens, suas desvantagens e a relação entre EDI e XML. O capítulo 3 apresenta conceitos de bancos de dados e descreve de forma sucinta banco de dados relacional e banco de dados orientado a objetos. O capítulo 4, por fim, mostra o estudo de caso realizado.

# 1 XML

A XML pode ser vista 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 para documentos utilizando *tags*, a XML oferece ao usuário a oportunidade de criar uma linguagem de marcação específica, de acordo com a sua necessidade.

O conceito de marcação criado por Ray (2001, p. 2) afirma que:

> Marcação é a informação incluída em um documento para melhorar seu significado, identificando as partes e como elas se relacionam umas com as outras. Uma linguagem de marcação é utilizada para formatar documentos, utilizando-se de marcadores (tags) para rotular e demarcar limites neste documento de modo que, ao processar, o programa de computador consiga distinguir uma parte do texto das outras.

Assim como documentos que incorporam outras linguagens de marcação, os documentos XML possuem a estrutura em forma de árvore, o que facilita a descrição do conteúdo e a formatação do documento. Além disso, este formato altamente estruturado torna mais fácil separar o conteúdo da apresentação e aplicar formatos específicos, de acordo com a finalidade, para um mesmo grupo de dados.

Sobre os documentos XML, Chang [*et al.*] (2001, p.270) explica que "a XML descreve uma classe de objetos de dados que tem estrutura aninhada e é usada para marcar (comentar) os dados contidos nesses objetos. Esses objetos são conhecidos como documentos XML e são, por definição, compatíveis com documentos SGML".

Os documentos XML serão mais bem descritos na seção 1.4.4.

## 1.1 ORIGEM E OBJETIVOS

A XML surgiu da necessidade de adaptar a SGML para a *Web*. Criada no final da década de 60 pelos pesquisadores C. Goldfarb, E. Mosher e R. Lorie, a SGML tornou-se um padrão internacional (ISO – 8879) em 1986, passando a ser utilizada por diversas empresas para a estruturação de documentos. Oferece um esquema de marcação simples, independente de plataforma e extremamente flexível. Justamente por ser bastante flexível, o software criado para processa-la é por demais grande e caro, sendo sua utilidade limitada a grandes organizações (RAY, 2001).

Antes mesmo da criação da XML, um tipo de documento SGML, compacto e eficiente, surgiu. O HTML foi desenvolvido em 1990 por Tim Berners-Lee e Anders Berglund, funcionários do CERN (Centro Europeu de Pesquisas Nucleares) de Genebra, para estruturar e apresentar documentos na *Web*. Esta linguagem tornou-se popular e passou a ser utilizada por milhares de aplicações, incluindo navegadores, editores, etc. Porém, a falta de extensibilidade e de uma estrutura claramente definida sinalizavam a necessidade de uma nova forma de manipular dados estruturados.

Reunindo a flexibilidade do SGML e a simplicidade do HTML, a primeira fase do desenvolvimento da XML foi registrada em novembro de 1996, pelo comitê W3C, grupo formado por empresas e organizações de diversas partes do mundo com o objetivo de discutir, promover e recomendar tecnologias *Web*. Em janeiro de 1997, surgiu o primeiro analisador sintático XML e alguns meses depois os primeiros aplicativos em XML começaram a aparecer. No final deste mesmo ano, foi implementado o suporte ao XML no navegador *Internet Explorer* da Microsoft. Finalmente, em fevereiro de 1998, a primeira recomendação XML foi publicada (PITTS-MOULTIS e KIRK, 2000).

De acordo com a recomendação do W3C de fevereiro de 1998, os objetivos definidos para a XML são:

- XML deve ser utilizada diretamente na Internet;
- XML deve suportar uma ampla variedade de aplicações;
- XML deve ser compatível com SGML;
- Deve ser fácil escrever programas que processem documentos XML;
- O número de recursos opcionais na XML deve ser mínimo ou zero;
- Os documentos XML devem ser legíveis e relativamente claros;
- O projeto da XML deve ser possível de ser preparado rapidamente;
- O projeto da XML deve ser formal e conciso;
- Os documentos XML devem ser facilmente criados;
- A concisão na marcação da XML é de mínima importância.

Estas especificações, juntamente com outras normas associadas (Unicode e ISO/IEC 10646 para caracteres, RFC 1766 para linguagem de marcação, ISO 639 para códigos de linguagem de nomes, ISO 3166 para os códigos de nomes de países) oferecem todas as informações necessárias para entender XML e para construir programas de computador para processá-la.

## 1.2 VANTAGENS

Após seu registro, em 1996, a XML vem despertando a cada dia um número maior de interessados em conhecer e utilizar sua maneira padrão de codificar o conteúdo e criar

estruturas de dados flexíveis. Apesar de ter seu surgimento associado à necessidade de trocas de informações eletrônicas via Internet, sua importância não se limita a isso. A adoção da XML vem invadindo, com destaque, a área de aplicações de banco de dados.

Dentre as suas vantagens, destacam-se:

- Simplicidade: A XML é intuitiva, simples e de fácil interpretação;
- Interoperabilidade: A utilização de documentos XML não depende de aplicação ou sistema operacional, tornando simples a integração entre sistemas heterogêneos;
- Flexibilidade: Permite manter informações de estilo fora do documento, através da utilização de folhas de estilo. Dessa forma, pode-se utilizar a mesma configuração de estilos para vários documentos, assim como modificá-la uma única vez mesmo que vários documentos a utilizem;
- Extensibilidade: permite a criação de *tags* específicas, de acordo com a necessidade do usuário;
- Padrão para visões de banco de dados: Fornece um padrão para expressar a estrutura de visões[1] de dados. Esta característica permite que bancos de dados capazes de interpretar a estrutura definida em um documento XML tenham a possibilidade de transmitir e processar visões.
- Integração de dados de diferentes fontes: Com a XML pode-se realizar facilmente uma busca em diferentes bancos de dados, combinando-os conforme necessário.

[1] "Uma visão é qualquer relação que não faz parte do modelo lógico, mas é visível ao usuário como uma relação virtual" (SILBERCHATZ; KORTH; SUDARSCHAN, 1999).

- Máxima verificação de erros: Os documentos XML são analisados por um *parser*[2] a fim de evitar erros. O *parser* analisa a grafia dos nomes de elementos e informa quando encontra algum objeto incorreto.
- Diferentes formas de visualização dos dados: A XML define somente os dados e não o visual. Por isso, os dados de um documento XML poderão ser visualizados de formas múltiplas, de acordo com a aplicação.

.

## 1.3 DESVANTAGENS

Apesar as vantagens citadas, a XML apresenta também algumas desvantagens com relação a sua utilização:

- Por ter formato texto, documentos XML consomem maior largura de banda[3] e espaço de armazenamento do que formatos binários. Desta mesma forma, p*arsers* XML podem apresentar baixa performance e por isso requerem mais memória;
- Documentos XML definem a estrutura dos dados, mas não descrevem como manipulá-los. Por esse motivo diz-se que sua semântica é limitada.

[2] Programa que lê e interpreta dados descritos em um documento XML.
[3] Largura de banda: quantidade de informação passível de ser transmitida por unidade de tempo, num determinado meio de comunicação.

## 1.4 ESTRUTURA DE APLICAÇÕES XML

Para desenvolver uma aplicação XML de forma a obter o máximo de benefícios disponíveis, é necessário estar atento aos componentes básicos, definidos na especificação XML, a serem utilizados em sua infra-estrutura. Dentre estes componentes, serão brevemente discutidos os Elementos, Atributos, Entidades e DTDs.

### 1.4.1 Elementos

Elementos são marcadores utilizados em um documento XML ou no interior de uma DTD para especificar a estrutura do documento e garantir maior flexibilidade para definir estruturas de acordo com uma necessidade específica. Os elementos fornecem a marcação para informar ao documento e ao processador o que ele deverá realizar (PITTS-MOULTIS e KIRK, 2000).

Diferente do HTML, onde o elemento é utilizado para descrever a marca, na XML os elementos possuem a função de descrever os dados. Além disso, um elemento especifica como manipular os dados contidos entre as marcas de início e fim.

Os elementos poderão ser mais facilmente entendidos através do exemplo a seguir:

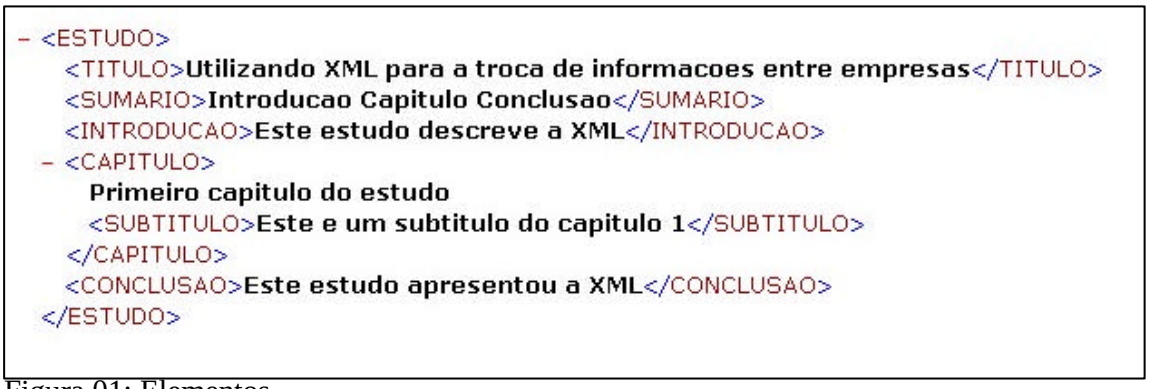



Figura 01: Elementos

Fonte: Próprio Autor

Neste exemplo, o elemento `<ESTUDO></ESTUDO >` armazena todo o conteúdo do documento. Os elementos `<TITULO>`, `<SUMARIO>`, `<INTRODUCAO>`, `<CAPITULO>` e `<CONCLUSAO>` são, na realidade, sub-elementos aninhados sob o elemento `<ESTUDO>`.

### 1.4.2 Atributos

Os atributos são utilizados para fornecer informações adicionais sobre um elemento XML e seu conteúdo. Ajudam a descrever exatamente o que são os elementos, o tipo de informação que deve ser inserido neles e a ordem na qual as informações devem der colocadas (PITTS-MOULTIS e KIRK, 2000).

De forma resumida, pode-se dizer que um atributo é utilizado para especificar propriedades ou características de um elemento. Os atributos são separados por espaços e devem aparecer após o nome do elemento ou em uma lista de atributos após todos os elementos terem sido declarados na DTD.

### 1.4.3 Entidades

As entidades são utilizadas para tornar a XML mais fácil de ser escrita, mantida e lida. É tida como um marcador de lugar declarado uma única vez e que pode ser utilizado diversas vezes, em diferentes partes do documento.

Faz-se necessário o uso de entidades, por exemplo, em situações onde a mesma informação é utilizada em diversas partes, evitando a duplicidade. Outros exemplos seriam a representação de caracteres impossíveis de serem digitados e a marcação do lugar onde um arquivo deverá ser importado (RAY, 2001).

O exemplo abaixo descreve a utilização de uma entidade para representar uma informação que deverá ser visualizada em diversas partes do documento:

```
<TESTE> XML - &descricao </TESTE>
```

Onde a saída apareceria conforme descrito a seguir:

```
<TESTE>XML - Extensible Markup Language</TESTE>
```

A sintaxe para declarar uma entidade é `<!ENTITY Nome “definição_entidade”>`.

### 1.4.4 DTD

Segundo Pitts-Moultis e Kirk (2000, p.33), “DTDs – *Document Type Definitions* (definições de tipos de documentos) são as estruturas que fornecem todas as ferramentas usadas para criar documentos XML”. De forma um pouco mais simples, a DTD é um arquivo que fornece regras para o documento XML ao qual ele está anexado.

Dentro de uma DTD são definidas as partes de um documento e são descritas como estas partes podem ou não ser utilizadas e se são ou não componentes obrigatórios do documento. Da mesma forma uma DTD pode incluir as declarações de elementos, atributos e entidades que se deseja utilizar.

Um documento XML que referencie uma DTD é analisado por um processador XML para verificar se as restrições descritas na DTD estão sendo violadas de alguma forma. Caso o processador considere o documento XML compatível com sua respectiva DTD, o documento é considerado válido. Documentos válidos são descritos no item 1.4.4.1 deste trabalho.

A DTD pode ser feita de 2 (duas) formas:

- DTDs Internas: As definições e regras são incluídas no documento em si. É indicado que este tipo de DTD seja utilizado em documentos que necessitem ser auto-suficientes, ou seja, possam ser movidos entre diferentes sistemas sem perder componentes e em situações onde existam restrições de tempo de processamento e de largura de banda;
- DTDs Externas: As definições e regras são armazenadas em um arquivo externo ao documento XML. Embora este tipo de DTD possa representar um aumento de tempo de processamento e de exigência de largura de banda, ele permite que o arquivo com as definições seja reutilizado, além de tornar o gerenciamento, as atualizações e as modificações mais fáceis.

A utilização de DTDs não se faz necessária para todo e qualquer documento XML. Para entender quando é necessário utilizá-la é importante conhecer os conceitos de documentos válidos e documentos bem formatados.

#### 1.4.4.1 Documentos válidos

De acordo com Pitts-Moultis e Kirk (2000, p.112), “Um documento XML válido é aquele cujo código e conteúdo estejam corretos e as regras do XML como as da DTD sejam seguidas”.

Para que um documento XML seja considerado válido, ele deve estar adequado a especificação XML e a DTD agregada ao documento. Neste caso é necessário haver uma definição de tipo de documento.

#### 1.4.4.2 Documentos bem formatados

Documentos são considerados bem formatados quando atendem as regras de sintaxe XML. Segundo a terceira edição da especificação XML, de 4 de fevereiro de 2004, um objeto textual é um documento bem formatado caso:

- Como um todo, ele coincida com o documento identificado como de produção;
- Atenda as restrições de boa formatação dadas na especificação acima;
- Cada uma das entidades analisadas sintaticamente que sofrem referência direta ou indireta no interior do documento seja bem-formatada.

## 1.5 TECNOLOGIAS ASSOCIADAS AO XML

### 1.5.1 XSL

Para entender o XSL (*Extensible Stylesheet Language*), é possível fazer uma analogia com documentos criados por meio do *Microsoft Word* (ou outro editor de textos comum). Ao criar um documento nesta ferramenta é possível definir estilos, ou seja, definir com que aparência será apresentada cada parte do texto (fonte, cabeçalho, cor, etc.) de forma que um estilo possa ser facilmente alterado. O conjunto de regras de formatação que compõem um estilo é chamado folha de estilo.

Segundo Pitts-Moultis e Kirk (2000, p. 347), "o XSL é o mecanismo usado para criar folhas de estilos para documentos XML", ou seja, o XSL é utilizado para fornecer informações sobre a estrutura de um documento através de folhas de estilo, instruindo o navegador ou software de exibição sobre como formatar o conteúdo do documento para a visualização do usuário.

### 1.5.2 DOM

De acordo com a especificação do W3C (terceira edição), o DOM (*Document Object Model*) é "a definição da estrutura lógica dos documentos e o meio pelo qual um documento é acessado e manipulado". Por meio de interfaces DOM, é possível criar documentos e navegar através de suas estruturas, bem como adicionar, modificar e excluir

elementos ou parte do conteúdo. De forma mais específica, o DOM pode ser utilizado para coletar informações sobre a estrutura de um documento XML e, por meio destas informações, criar conexões entre a estrutura do documento e outros aplicativos ou bases de dados.

No DOM, os documentos possuem uma estrutura lógica em formato de árvore que possibilita ao usuário recuperar rapidamente as informações ou componentes XML que se deseja. Dentro desta estrutura de árvore, tudo é visto como um objeto, desde o nó-raiz até o valor da entidade. A figura a seguir ilustra a forma como o DOM representa um documento:

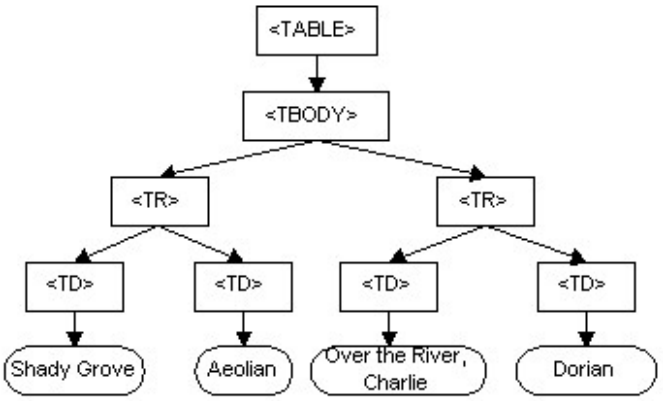



Figura 02 – Representação DOM.

Fonte: www.w3c.org.

Para acessar um arquivo XML, o DOM utiliza três objetos:

- Documento XML: Consiste do elemento-raiz juntamente com todos os seus descendentes;
- Nó XML: Representa um nó dentro de um documento XML;
- Lista de nós XML: Conjunto de nós XML.

### 1.5.3 XPath

O objetivo principal do XPath é direcionar partes de um documento XML. Como suporte a este objetivo principal, o XPath também proporciona funções básicas para o tratamento de *strings*, números e valores *booleanos*.

Utilizando expressões de caminho que especificam um arquivo dentro de um diretório, o XPath define como será realizado o acesso aos nós dentro de um documento de forma facilitada. Quando uma expressão é avaliada, o resultado é um objeto do tipo *node-set* (conjunto não organizado de nós sem duplicata), *boolean* (verdadeiro ou falso), *number* ou *string*.

O XPath utiliza uma sintaxe não XML e compacta para facilitar a sua utilização em URLs[4] e em valores de atributos.

## 1.6 CONCLUSÃO

A XML é uma meta linguagem de marcação que permite a estruturação de dados e a integração entre diferentes sistemas de informação. Surgiu de forma a reunir as vantagens do SGML e do HTML, superando as limitações impostas por essas tecnologias. Dentre as diversas vantagens que justificam a crescente adoção desta tecnologia, destacam-se: simplicidade, flexibilidade, extensibilidade, integração entre dados de diferentes fontes, máxima verificação de erros e diferentes formas de visualização dos dados.

[4] Identifica o endereço de um domínio na Internet

## 2 EDI

Segundo Tubino (p.170), “o EDI (*Eletronic Data Interchange*) é uma tecnologia baseada em plataformas de hardware e software, que permite a troca de documentos de negócios (faturas, duplicatas, pedidos de compra, avisos de despacho, *kanbans*, etc.) eletronicamente e de forma padronizada entre duas empresas distintas, situadas em diferentes locais”.

O surgimento do EDI aconteceu na década de 60, nos Estados Unidos, decorrente do trabalho de associações comerciais e setoriais. Na indústria automobilística a adoção do EDI foi feita visando obter maior eficiência na produção, já as companhias aéreas passaram a utilizar este processo para controle e reserva de passagens.

O EDI trabalha com um formato estruturado de dados, seguindo um padrão estabelecido entre remetente e receptor, de forma que os documentos a serem trocados possam ser reconhecidos em termos de conteúdo, significado e formato, permitindo seu processamento automático e isento de ambigüidades. O uso de códigos compactos e de estruturas de dados concisas resulta em mensagens de EDI quase impossíveis de serem interpretadas por pessoas sem o auxílio de uma documentação.

A criação de normas para transações de negócio permitiu eliminar barreiras na comunicação entre organizações, tornando a troca de informações rápida e eficiente. O EDI permite que documentos sejam enviados entre empresas através de uma rede de dados e com isso possibilita reduzir o número de erros por re-digitação e o volume de papéis. Além disso, o EDI permite às empresas um melhor gerenciamento da cadeia logística e

conseqüentemente um ganho de produtividade através da reposição contínua de materiais (EANBRASIL, 05/09/2004).

O EDI pode ser classificado de acordo com 2 (duas) categorias:

- EDI Tradicional: A transferência de arquivos é realizada seguindo um padrão estabelecido entre remetente e receptor, de modo que o formato do documento seja compatível com a aplicação utilizada na origem e no destino. Uma das maiores limitações desta categoria de EDI é o alto custo de implementação;
- Web-EDI: A tecnologia *Web* é utilizada para realizar a transferência de arquivos. Através de formulários preenchidos em um *browser* na Internet, a informação é transformada em mensagem EDI e encaminhada ao destino.

## 2.1 VANTAGENS

- Diminuição no tempo: Dentro do processo de EDI os documentos são transmitidos e processados de forma eletrônica e sem a necessidade de intervenção manual. Com isso, o tempo para realização destas tarefas é reduzido;
- Diminuição de Erros: A utilização de sistemas de informação alinhada a novos métodos de comunicação assegura a integridade dos dados e reduz de forma significativa a possibilidade de ocorrerem problemas de métodos de transmissão de documentos anteriormente utilizados;

- Redução de Custos Administrativos: Utilizando-se da transmissão eletrônica de documentos evita-se a necessidade de re-introduzir os dados. Desta forma, os recursos humanos e técnicos anteriormente envolvidos neste processo tornam-se disponíveis para outras tarefas;

## 2.2 DESVANTAGENS

- Custo elevado: alto custo de desenvolvimento e implementação;
- Alto grau de adaptabilidade: incorpora rígidas regras comerciais;
- Falta de flexibilidade: O EDI é uma solução ponto a ponto que deve receber novo trabalho de engenharia toda vez que a empresa agregar um novo parceiro comercial.

## 2.3 DO EDI AO XML

O EDI surgiu com o objetivo de suprir a necessidade de troca de informações eletrônicas dentro da mesma organização ou entre organizações diferentes. As diversas limitações impostas por esse processo, conforme descrito no na seção 2.2 deste capítulo, deram origem a um movimento crescente em direção a converter os sistemas EDI para sistemas que utilizem a tecnologia XML.

A troca eletrônica de dados permite que a comunicação entre organizações seja feita de forma bidirecional, porém com alto custo e com rígidas regras incorporadas. Além disso,

para cada parceiro, cliente ou organização com a qual deseja-se estabelecer uma comunicação via EDI é necessário um alto grau de adaptabilidade, exigindo-se normalmente uma customização do sistema.

Com todas estas características desfavoráveis ao EDI, as organizações partiram em busca de um padrão mais econômico e menos complexo para o intercâmbio de informações. Com isso, teve início a adoção da XML por estas organizações, não apenas por suprir as necessidades citadas, mas também pelo número significativo de vantagens oferecidas.

Com a utilização da XML, a comunicação empresarial através da Internet tornou-se muito mais simples. Além disso, os mecanismos de transformação de dados da XML ultrapassam barreiras anteriormente impostas por sistemas operacionais, bases de dados e aplicativos tornando-o ideal para ambientes heterogêneos.

## 2.4 CONCLUSÃO

O EDI é a transferência de documentos de negócios, seguindo um padrão acordado entre emissor e receptor, por meio eletrônico. Durante vários anos o EDI representou vantagens como redução de tempo, de erros e de custos administrativos para as empresas que adotaram esta solução. Porém, o alto grau de adaptabilidade exigido, a falta de flexibilidade e os custos elevados envolvidos na implementação levaram as organizações à busca de um novo padrão para o intercâmbio de informações comerciais.

# 3 BANCO DE DADOS

Conforme Kroenke (1998, p.12), “um banco de dados é uma coleção autodescritiva de registros integrados”. A partir deste conceito, entende-se que um banco de dados possui a definição da sua própria estrutura, chamada dicionário de dados, o que possibilita determinar o seu conteúdo examinando-o, ou seja, sem a necessidade de uma documentação externa de formatos de registros. Além da sua própria descrição, um banco de dados contém arquivos de dados do usuário, índices e estruturas de formulários, originando uma coleção de registros integrados.

De forma simples, pode-se dizer que um banco de dados é um conjunto de dados armazenados de forma que possam ser recuperadas quando necessário. Para recuperar e armazenar de forma eficiente estas informações utiliza-se um Sistema Gerenciador de Banco de Dados (SGBD).

Um Sistema Gerenciador de Banco de Dados tem como objetivo definir as estruturas de armazenamento das informações e os mecanismos para manipulação das mesmas, além de garantir a segurança e impedir tentativas de acessos não autorizados às informações armazenadas (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

## 3.1 BANCO DE DADOS RELACIONAL

A idéia de banco de dados relacional surgiu em 1970, quando E. F. Codd publicou seu trabalho entitulado *“A Relational Model of Data for Large Shared Databanks*”, no qual conceitos de álgebra eram utilizados para resolver problemas de armazenamento de grandes

quantidades de dados. A partir desta publicação, iniciou-se o movimento que definiu o modelo de banco de dados relacional (KROENKE, 1998). Em 1976, uma nova e importante percepção do modelo relacional foi proposta por Peter Chen através do modelo Entidade-Relacionamento (ER), possibilitando ao projetista concentrar-se na utilização dos dados e não mais nas estruturas lógicas de tabelas.

(SQLMAGAZINE, 11/10/2004)

O modelo de banco de dados relacional utiliza um conjunto de relações para representar seus dados. Essas relações são chamadas de tabelas e armazenam informações sobre os dados e seus relacionamentos. Cada linha de uma tabela contém dados que pertencem a alguma coisa ou a uma parte desta. Cada coluna de uma tabela representa um conjunto de valores que devem ser de um mesmo tipo de dado. Cada um destes valores é representado por um campo. O conjunto de valores que um campo pode representar é chamado de domínio. A instância de uma tabela consiste no conjunto de valores que cada campo assume em um determinado instante. A existência de tabelas duplicadas, ou seja, com mesmo nome, não é permitida dentro do banco de dados. A figura abaixo ilustra uma relação:

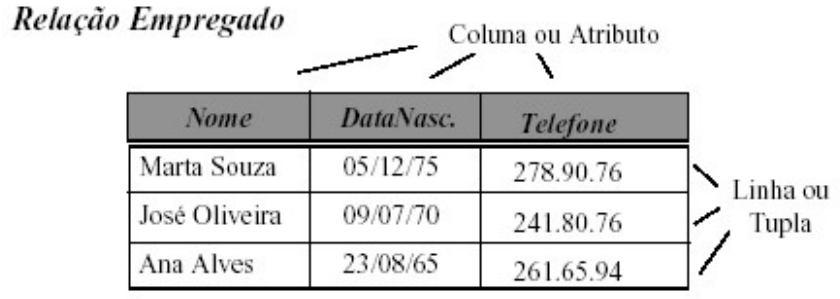



Figura 03: Exemplo de relação.

Fonte: http://www.lia.ufc.br/~eti/menu/modulos/BDR/BDR-ModeloRelacional.pdf

Além dos conceitos já abordados, algumas características devem fazer parte de um banco de dados relacional, entre elas:

- Controle de Redundância: A redundância consiste no armazenamento de uma mesma informação em locais diferentes. Em um banco de dados deve-se evitar este tipo de duplicação, armazenando as informações em um único local;
- Controle de Integridade: Impedir que aplicações ou acessos por meio de interfaces possam comprometer a integridade dos dados armazenados no banco de dados;
- Geração de visões: O banco de dados deve permitir aos usuários visualizar os dados de diferentes formas, não apenas da forma previamente existente no banco de dados;
- Suporte a tipos de dados: Todo banco de dados precisa prover suporte a diferentes tipos de dados;
- Interfaces de acesso: Disponibilizar formas de acesso aos dados contidos no banco de dados, seja por meio de interfaces gráficas, por linguagem SQL ou por qualquer outra forma;
- Suporte a procedimentos: Procedimentos são rotinas escritas em linguagem SQL armazenadas com o objetivo de automatizar tarefas. Para a execução destes procedimentos utiliza-se chamadas explícitas realizadas a partir de um sistema gerenciador de banco de dados;

A maior parte dos bancos de dados relacionais atualmente disponíveis no mercado utiliza a SQL como linguagem de consulta. Apesar de ser considerada uma linguagem de consulta , a SQL possui muitos outros recursos além da consulta ao banco de dados, como meios para definição da estrutura de dados, para modificação de dados no banco de dados

e para a especificação de restrições de segurança (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

### 3.1.1 Microsoft SQL Server 2000

O SQL Server é um sistema gerenciador de banco de dados relacionais que pode ser utilizado unicamente sob sistemas operacionais Microsoft Windows. Originalmente, o SQL Server foi desenvolvido baseado no Sybase SQL Server X versão 4.2. Posteriormente, na versão 06, a Microsoft implementou modificações visando fazer uso de características multitarefa do sistema operacional Windows NT e desenvolveu melhorias até chegar à versão 2000[5] (SQL MAGAZINE, 08/11/2004).

Dentre as características do SQL Server 2000, destacam-se:

- Ferramentas práticas de *Business Intelligence* (BI): Cada cópia do SQL Server 2000 acompanha um conjunto de serviços de BI;
- Capacidade de auto-direcionamento e gerenciamento: Recursos dinâmicos de auto-desempenho e auto-configuração otimizam o desempenho do banco de dados enquanto ferramentas de desenvolvimento automatizam atividades padrão. Ferramentas e assistentes gráficos simplificam a configuração, o *design* do banco de dados e o monitoramento do desempenho, permitindo aos administradores do banco de dados focar as necessidades estratégicas do negócio;
- Aplicações e serviços de gerenciamento de dados: Cada licença de software inclui ferramentas detalhadas de gerenciamento e desenvolvimento, uma ferramenta de

[5] Versão 2005 em fase de lançamento.

extração, transformação e carga, serviços de análise para soluções de BI e serviços de notificação;

- Suporte a múltiplas instâncias: Permite executar múltiplas instâncias do SQL Server na mesma máquina.

3.1.1.2 Suporte para XML no SQL Server 2000

O SQL Server 2000 oferece suporte integrado XML de forma flexível, de alto desempenho e fácil de ser utilizado. Por meio de recursos que permitem importar e exportar dados no formato XML, assim como integrá-los com um servidor *Web (Internet Information Service)* para envio de instruções SQL por meio de um navegador de internet, o SQL Server 2000 propõe uma forma de facilitar a tarefa de desenvolvedores da *Web* e de banco de dados através da utilização de tecnologias como o *XPath*, consultas via URL e XML.

Dentre os recursos disponíveis no SQL Server 2000 que facilitam a utilização da tecnologia XML, destacam-se:

- Acesso a consultas SQL via URL: é possível acessar uma consulta no SQL Server 2000 por meio de uma URL que utilize o protocolo HTTP;
- Visão relacional dos dados XML utilizando-se do *OpenXML*: Permite consultar arquivos XML e atualizar informações do arquivo no banco de dados;
- Resultado de consultas em formato XML: A utilização da cláusula FOR XML no SQL Server permite obter o resultado de uma consulta em formato XML;

- Capacidade de armazenamento de dados no formato XML: De forma eficiente, o SQL Server permite armazenar os dados como XML, mantendo o relacionamento e a hierarquia dos dados.
- XML *Updategrams*: Recurso que permite especificar uma atualização para um banco de dados do SQL Server 2000 através do XML;
- Utilização de *templates* contendo consultas pré-definidas: Desta forma as aplicações poderão consultar apenas os *templates* para obter o resultado esperado.

Ao executar uma consulta, de forma a estruturar o resultado no formato de um documento XML, o SQL Server oferece três modos distintos de visualização: *RAW, AUTO* e *EXPLICIT*. Estes modos serão descritos e exemplificados a seguir:

- *RAW*: Neste modo, cada registro é uma *tag* e cada campo do registro é um atributo da *tag*.

  Exemplo:

```
SELECT cd_funcionario, nm_funcionario
FROM funcionario
FOR XML RAW;
```

  Retorna:

```
<row cd_funcionario="1" nm_funcionario="Joao"/>
<row cd_funcionario="2" nm_funcionario="Maria"/>
<row cd_funcionario="3" nm_funcionario="Sergio"/>
<row cd_funcionario="4" nm_funcionario="Lucia"/>
```

.

- *AUTO*: No modo *AUTO*, cada registro é uma *tag* cujo nome é o mesmo da tabela. Cada valor do registro é um atributo da *tag* anteriormente citada. Ao utilizar este modo, é possível selecionar as seguintes opções:
    - *XMLDATA*: Juntamente com os dados, retorna informações sobre a tabela chamada *schema*. Estes *schemas* seguem o mesmo padrão já utilizado pela Microsoft;
    - *ELEMENTS*: Utilizando esta opção, para cada registro retornado da consulta, uma *tag* é aberta e fechada. Dentro desta *tag*, cada campo é retornado como uma outra *tag*, cujo valor é o próprio conteúdo do campo armazenado na tabela.

Exemplo:

```
SELECT cd_funcionario, nm_funcionario
FROM funcionario
FOR XML AUTO;
```

Retorna:

```
<funcionario cd_funcionario="1" nm_funcionario="Joao"/>
<funcionario cd_funcionario="2" nm_funcionario="Maria"/>
<funcionario cd_funcionario="3" nm_funcionario="Sergio"/>
<funcionario cd_funcionario="4" nm_funcionario="Lucia"/>
```

- *EXPLICIT*: O modo *EXPLICIT* é mais flexível e mais complicado de utilizar do que os modos *RAW* e *AUTO*. As consultas executadas neste modo retornam um documento XML em formato de uma tabela universal.

  A tabela universal é um mecanismo utilizado para retornar resultados de consultas no SQL Server 2000 que descreve como se deseja visualizar o documento. O

resultado da consulta é marcado com uma coluna especial de cabeçalho que mostra ao servidor como processar o documento XML.

Exemplo:

```
SELECT 1 as tag,
       null as parent,
       cd_funcionario as [funcionario!1!cd_funcionario],
       nm_funcionario as [funcionario!1]
FROM funcionario
FOR XML EXPLICIT;
```

Retorna:

```
<funcionario cd_funcionario="1">Joao</funcionario>
<funcionario cd_funcionario="2">Maria</funcionario>
<funcionario cd_funcionario="3">Sergio</funcionario>
<funcionario cd_funcionario="4">Lucia</funcionario>
```

### 3.1.2 Oracle 9i

A *Oracle Corporation* tem se mostrado pioneira no desenvolvimento de tecnologias para Internet. Um exemplo disso é o Oracle 9i, considerado uma evolução das tecnologias de banco de dados devido aos recursos disponíveis para atender as necessidades que surgiram com a ascensão do uso da Internet.

Dentre os vários aspectos que caracterizam o Oracle 9i como uma boa opção para Internet, destacam-se:

- Disponibilidade: Possui uma série de recursos para garantir a disponibilidade do serviço, como: proteção contra falhas, manutenção de dados *on-line* e correção de erros *self service*;

- Escalabilidade: Dispõe de uma solução de compartilhamento de disco (*Real Application Cluster*) que possibilita a várias máquinas acessar o mesmo banco de dados. Do mesmo modo, caso um dos servidores venha a falhar, os demais servidores continuam funcionando de forma transparente;
- Segurança: A criação de três camadas de segurança no banco de dados permite que o mesmo não seja acessado de forma direta, somente por meio de uma aplicação. Além disso, pode ser implantada criptografia via chave pública ou SSL (*Secure Sockets Layer*), criar perfis de usuários, entre outros recursos;
- Inteligência de negócios: Facilita o processo de análise para geração de *data warehouse*[6] e *data mining*[7], permitindo minimizar o custo de implantação destas soluções;
- Plataforma de desenvolvimento: Permite uma escolha confortável da plataforma de desenvolvimento, possui suporte a XML e também a plataforma tradicional de desenvolvimento da Oracle em SQL e PL/SQL.

#### 3.1.2.1 Suporte para XML no Oracle 9i

O SGBD Oracle, a partir da versão 9i, apresenta um novo tipo de objeto, o *XML Type*, que oferece mecanismos para criar, extrair e indexar dados XML. Com o *XMLType*,

[6] Coleção de dados orientada por assuntos, integrada, variante no tempo e não volátil, que tem por objetivo dar suporte aos processos de tomada de decisão.

[7] Processo analítico projetado para explorar grandes quantidades de dados de forma a identificar padrões válidos, novos, potencialmente úteis e ultimamente compreensíveis.

desenvolvedores podem utilizar todos os recursos oferecidos por um banco de dados relacional e simultaneamente trabalhar em um contexto XML.

Entre os benefícios oferecidos pelo *XMLType*, destacam-se:

- A "união" entre XML e SQL: Possibilita operações SQL em contexto XML e operações XML em contexto SQL;
- Funções pré-definidas: Possui funções pré-definidas para indexação, navegação, entre outras;
- Melhor performance e escalabilidade: O *XMLType* utiliza um *parser* e um processador XML embutidos, de forma a obter melhor performance e escalabilidade;
- Permite combinar comandos SQL com outros tipos de dados: Como exemplo, pode-se executar uma consulta a colunas *XMLType* juntar o resultado com o resultado de uma consulta a uma coluna VARCHAR;
- Permite criar índices nas funções *existsNode()* e *extract()*: Desta forma é possível aumentar a velocidade de processamento da consulta.

O Oracle 9i reúne recursos que simplificam a manipulação de documentos XML (*Oracle XML Parser*) e a transformação desses documentos em HTML, texto ou outra estrutura (*Oracle XML Processos*). Para facilitar a criação de um documento XML a partir dos resultados obtidos de uma consulta executada no banco de dados e tornar simples o armazenamento deste documento em tabelas no banco, o Oracle 9i possui o *Oracle XML SQL Utility.*

## 3.2 BANCO DE DADOS ORIENTADO A OBJETO

O conceito de banco de dados orientado a objeto teve origem a partir de restrições impostas pelo modelo relacional que limitaram a utilização de algumas aplicações que utilizavam tipos de dados mais complexos como hipertexto, vídeo, imagem e som.

Segundo Kroenke (1998, p. 313), “um objeto é uma estrutura encapsulada que tem tanto atributos como métodos”. A partir deste conceito, entende-se que o banco de dados orientado a objeto trabalha com o conceito de encapsulamento para esconder detalhes da implementação de métodos e da estrutura de atributos. Dessa forma, garante que atributos de uma determinada classe sejam acessados via métodos.

No sistema de gerenciamento de banco de dados orientado a objetos, a idéia de objeto é utilizada no nível lógico e possui características para atender as necessidades das linguagens de programação orientadas a objetos, como operadores de manipulação de estruturas, gerenciamento de armazenamento, tratamento de integridade e persistência dos dados.

## 3.3 CONCLUSÃO

A crescente aceitação da XML e seu uso cada vez maior tende a gerar uma grande quantidade de dados neste formato. Alguns deles são transientes e não necessitam ser armazenados, porém muito desses dados representam informações importantes e que precisam ser armazenadas. Em função disso, a tecnologia de bancos de dados vem sofrendo modificações para se adequar ao padrão XML. A necessidade de integrar documentos XML

com modelos de dados relacionais já reflete nas principais empresas que desenvolvem sistemas gerenciadores de bancos de dados, como a Oracle e a Microsoft, que estão investindo na construção de versões de seus respectivos produtos para atender a essa nova demanda.

# 4 ESTUDO DE CASO

## 4.1 A EMPRESA

A Busscar Ônibus S.A. foi fundada em 1946 como uma marcenaria que comercializava móveis, esquadrias e balcões de madeira. Um ano depois, a Nielson & Irmão (razão social na época) reformou a primeira carroceria de ônibus. Em 1949 foi montada a primeira estrutura totalmente de madeira, sobre um chassi Chevrolet Gigante. Hoje, a Busscar está entre as maiores fabricantes de carrocerias de ônibus da América Latina.

O Grupo Busscar atualmente é composto pelas filiais Busscar Plásticos, atuante no ramo de materiais plásticos, Climabuss, fabricante de ar condicionado e Busscar Rio Negrinho. A empresa possui ainda filiais na Bahia, Goiás, Minas Gerais, Paraná, Pernambuco, Rio de Janeiro e São Paulo. Além disso, conta com unidades industriais em diversos países, como México, Noruega, Colômbia, Venezuela e Cuba.

A divisão de Pesquisa e Desenvolvimento da empresa é fortemente apoiada pela área de Engenharia, com suas estações de CAD[8] (*Computer Aided Design*), tanto para lançamentos de novos produtos como também para as adaptações nos projetos já existentes no mercado. A política de qualidade conferida pela certificação ISO 9001, obtida no ano de 1997, se mostra presente em cada fase dos processos de fabricação e serviços da empresa. Sempre disposta a novos desafios, a empresa desenvolve projetos tanto nas linhas urbana quanto rodoviária, além de projetos especiais.

[8] Software específico para gerar desenhos técnicos em computadores.

Melhorar cada vez mais a gestão e aumentar a rentabilidade do negócio é a principal estratégia que sustenta a atuação da Busscar no mercado. Por isso, nos últimos anos a empresa fortaleceu-se no Brasil e conquistou maior participação no mercado internacional.

Nos próximos tópicos será apresentado o cenário atual em relação aos sistemas de informação utilizados na empresa.

## 4.2 CENÁRIO ATUAL

A Busscar possui uma equipe de profissionais de Tecnologia da Informação responsável pelo desenvolvimento de um sistema corporativo próprio, onde cada módulo é projetado e desenvolvido de acordo com as necessidades específicas da empresa e as informações são armazenadas em uma base de dados Oracle.

A filial Busscar Plásticos utiliza os módulos Ativo Imobilizado, Compras, Contábil, Estoque, Faturamento, Financeiro, Livros Fiscais e PCP do sistema Protheus, desenvolvido em linguagem AdvPL (*Advanced Protheus Language*) pela empresa Microsiga Software S.A. As informações fiscais cadastradas e processadas via sistema Protheus são armazenadas em um banco de dados Microsoft SQL Server 2000. A comunicação entre matriz e filial é feita através de antenas microondas que fornecem um link dedicado de até 11Mb.

Para que as informações fiscais da filial fossem transferidas, por meio eletrônico, para a base de dados da matriz, foi implementado, no início do ano de 2002, o processo de EDI. Neste processo, a matriz acessa um módulo customizado do sistema Protheus e, por

meio deste, gera um arquivo do tipo texto com as informações necessárias, respeitando um *layout* previamente definido (anexo 01).

Para que as informações gravadas no arquivo sejam incorporadas ao banco de dados Oracle, a matriz utiliza rotinas específicas para cada filial. No caso da filial Busscar Plásticos, o processo de importação inicia-se com uma tela do sistema da matriz (anexo 02), onde o arquivo é lido e validado. Após a execução desta rotina, as informações ficam armazenadas em tabelas temporárias até que seja realizado o processo chamado “Aceite de Notas Fiscais” (anexo 03).

## 4.3 PROPOSTA

De forma a demonstrar a aplicação dos conceitos descritos nos capítulos anteriores, desenvolveu-se a proposta abaixo de modo a efetuar a transferência de informações entre matriz e filial por meio de documentos XML, substituindo o processo de EDI atualmente utilizado. Esta proposta foi desenvolvida utilizando-se apenas das informações do cabeçalho da nota fiscal.

### 4.3.1 Definição da DTD

Conforme descrito na seção 1.4.4 do primeiro capítulo deste estudo, a DTD é utilizada para definir as partes do documento XML, bem como essas partes poderão ser utilizadas e se são ou não componentes obrigatórios. Considerando isso, foi definido o DTD para o cabeçalho da nota fiscal:

```
<?xml version="1.0"?>

<!ELEMENT cabecalho (identificador, nr_nota, nr_serie,
dt_emissao, cfo, vl_tot_nota, base_ipi, base_icms, vl_ipi,
vl_icms, vl_mercadoria, cgc, cgc_estab, cgc_digito, vl_seguro,
vl_desp_acessorios)>

<!ELEMENT identificador      (#PCDATA)>
<!ELEMENT nr_nota            (#PCDATA)>
<!ELEMENT nr_serie           (#PCDATA)>
<!ELEMENT dt_emissao         (#PCDATA)>
<!ELEMENT cfo                (#PCDATA)>
<!ELEMENT vl_tot_nota        (#PCDATA)>
<!ELEMENT base_ipi           (#PCDATA)>
<!ELEMENT base_icms          (#PCDATA)>
<!ELEMENT vl_ipi             (#PCDATA)>
<!ELEMENT vl_icms            (#PCDATA)>
<!ELEMENT vl_mercadoria      (#PCDATA)>
<!ELEMENT cgc                (#PCDATA)>
<!ELEMENT cgc_estab          (#PCDATA)>
<!ELEMENT cgc_digito         (#PCDATA)>
<!ELEMENT vl_seguro          (#PCDATA)>
```

Figura 04: DTD.

Fonte: Próprio autor.

4.3.2 Geração do documento XML no SQL Server 2000

Inicialmente foi criado um arquivo no formato texto (`C:\temp\root.txt`) que será utilizado para inserir a linha inicial do documento e o elemento root (*tag* que contém todo o conteúdo do documento), já que o *FOR XML* por si só não gera estas linhas.

```
<?xml version="1.0"?>
<!DOCTYPE cabecalho SYSTEM "C:\temp\cabecalho.dtd">
<ROWSET>
<%begindetail%>
<%insert_data_here%>
<%enddetail%>
</ROWSET>
```

Figura 05: Inserir elemento root.

Fonte: Próprio autor.

Conforme mostra a figura acima, foram utilizadas *tags* especiais que serão interpretadas pelo SQL Server, onde:

`<%begindetail%>`: Indica a leitura de dados que estão no formato de tabela;

`<%insert_data_here%>`: Efetua a leitura do próximo campo;

`<%enddetail%>`: Indica o término de linhas da tabela.

O SQL Server possui um procedimento (*stored procedure*) chamado *sp_makewebtask* que pode ser utilizado para executar consultas e para gerar arquivos HTML. A clausula *FOR XML* com o *SELECT* foi utilizada em conjunto com o

procedimento *sp_makewebtask* para criar o arquivo XML, conforme mostra a figura abaixo:

```
EXEC dbo.sp_makewebtask
 @outputfile = 'c:\temp\cabecalhonf.xml',
 @query = 'SELECT * FROM mt_nota_fiscal_rc FOR XML AUTO,
ELEMENTS',
 @templatefile ='c:\temp\root.txt'
```

Figura 06: Procedimento *sp_makewebtask*.

Fonte: Próprio autor.

Embora o método acima seja uma das formas mais fáceis de exportar dados relacionais para o formato XML, existe a necessidade de permissão especial para a execução do procedimento *sp_makewebtask*.

Como resultado do comando descrito na figura 06, obteve-se o documento XML desejado:

```
<?xml version="1.0"?>
<!DOCTYPE cabecalho SYSTEM "C:\temp\cabecalho.dtd">
<ROWSET>
     <ROW>
          <identificador>1</identificador>
          <nr_nota>1</nr_nota>
          <nr_serie>1</nr_serie>
          <dt_emissao>29-oct-2004</dt_emissao>
          <cfo>1111</cfo>
          <vl_tot_nota>10.5000</vl_tot_nota>
          <base_ipi>0.4000</base_ipi>
          <base_icms>0.10000000</base_icms>
          <vl_ipi>0.1000</vl_ipi>
          <vl_icms>0.1000</vl_icms>
          <vl_mercadoria>10.5000</vl_mercadoria>
          <cgc>1111111111</cgc>
          <cgc_estab>11111</cgc_estab>
          <cgc_igito>111</cgc_igito>
          <vl_seguro>0.0000</vl_seguro>
```

Figura 07: Documento XML.

Fonte: Próprio autor.

### 4.3.3 Importação do documento XML no Oracle 9i

Os procedimentos utilizados a fim de importar os dados do documento XML para dentro de uma tabela no banco de dados Oracle 9i serão descritos a seguir:

- Criação de um objeto diretório para mapear o diretório físico que contém o documento XML;

```
CREATE DIRECTORY XML_DIR AS '/tmp'
```

- Criação de uma tabela contendo o BFILE[9] e inserção de um registro para o documento XML, conforme mostra a figura a seguir:

```
CREATE TABLE XML_TEMP (key NUMBER, f_lob BFILE);
```

```
INSERT INTO XML_TEMP
VALUES (1,BFILENAME('XML_DIR','cabecalhonf.xml'));
```

Figura 08: Tabela XML_TEMP.

Fonte: Próprio autor.

- Criação de uma tabela para armazenar os dados contidos no documento XML. É importante lembrar que o nome das colunas deve ser exatamente igual as *tags* XML. A figura abaixo descreve a tabela MT_NOTA_FISCAL_RC:

```
create table MT_NOTA_FISCAL_RC
    (IDENTIFICADOR varchar2(1)
```

[9] Contém dados binários armazenados fora do banco de dados em arquivos do sistema operacional.

```
    ,NR_NOTA number(8)
    ,NR_SERIE varchar2(3)
    ,DT_EMISSAO varchar2(20)
    ,CFO varchar2(4)
    ,VL_TOT_NOTA number(16,4)
    ,BASE_IPI number (16,4)
    ,BASE_ICMS number(16,4)
    ,VL_IPI number(16,4)
    ,VL_ICMS number(16,4)
    ,VL_MERCADORIA number(16,4)
    ,CGC number(10)
    ,CGC_ESTAB number(5)
    ,CGC_DIGITO number(3)
    ,VL_SEGURO number(16,4)
    ,VL_DESP_ACESSORIOS number(16,4));
```

Figura 09: Tabela MT_NOTA_FISCAL_RC.

Fonte: Próprio autor.

- Desenvolvimento do procedimento para efetuar a importação do documento na tabela do banco de dados;

Este procedimento foi desenvolvido utilizando-se do pacote DBMS_LOB. Este pacote possui um conjunto de rotinas que visam facilitar a manipulação de lobs. Os lobs (*Large Objects*) são tipos de dados que podem armazenar informações de até 4 GB de dados binários ou caracteres.

A figura 10 apresenta o procedimento utilizado:

```
CREATE OR REPLACE PROCEDURE loadxml (p_key number, p_tabela
varchar2) AS
fil    BFILE;
buffer RAW(32767);
len    INTEGER;
insrow INTEGER;
BEGIN

SELECT f_lob
```

```
INTO fil
FROM xml_temp
WHERE key = p_key;
DBMS_LOB.FILEOPEN(fil,DBMS_LOB.FILE_READONLY);
len := DBMS_LOB.GETLENGTH(fil);
DBMS_LOB.READ(fil,len,1,buffer);
xmlgen.resetOptions;
insrow :=
xmlgen.insertXML(p_tabela,UTL_RAW.CAST_TO_VARCHAR2(buffer));
DBMS_OUTPUT.PUT_LINE(insrow);
IF DBMS_LOB.FILEISOPEN(fil) = 1 THEN
   DBMS_LOB.FILECLOSE(fil);
END IF;
EXCEPTION
WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE(sqlerrm);
     DBMS_OUTPUT.PUT_LINE(SQLERRM(SQLCODE));
IF DBMS_LOB.FILEISOPEN(fil) = 1 THEN
   DBMS_LOB.FILECLOSE(fil);
END IF; end;
```

Figura 10: Procedimento *loadxml.*

Fonte: Próprio autor.

- Para obter o resultado, o procedimento foi executado no SQL*Plus, conforme a figura abaixo:

```
SQL> set serveroutput on
SQL> exec loadxml(1,'mt_nota_fiscal_rc';

SQL> select * from mt_nota_fiscal_rc;
```

Figura 11: Execução do procedimento *loadxml*.

Fonte: Próprio autor.

- O resultado obtido foi à inserção dos dados contidos no documento XML na tabela MT_NOTA_FISCAL_RC, conforme demonstra a figura 12.

```
SQL> select * from mt_nota_fiscal_rc;
```

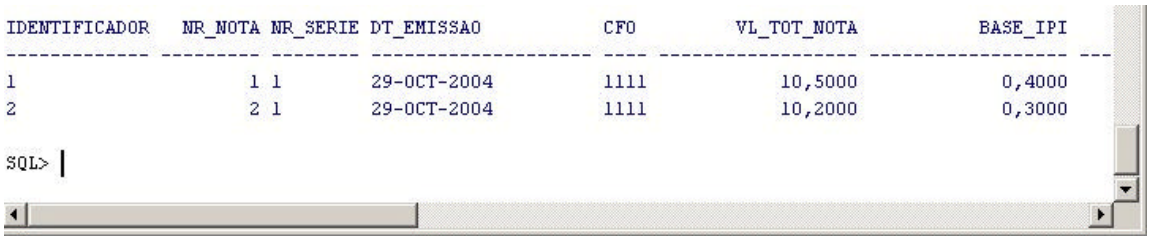



Figura 12: Registros inseridos na tabela MT_NOTA_FISCAL_RC.

Fonte: Próprio autor.

## 4.4 CONCLUSÃO

Uma das grandes vantagens de se utilizar XML na troca de informações entre empresas é a extensa relação de bibliotecas e procedimentos que as linguagens de programação e os bancos de dados vêm oferecendo. Especialmente nos bancos de dados utilizados no estudo de caso são fornecidos diversos recursos para permitir leitura, manipulação e escrita eficiente de arquivos XML.

A complexidade encontrada no processo de EDI para se adicionar um novo parceiro comercial pode ser facilmente superada com a utilização da XML. O arquivo texto com as especificações do intervalo de posições que corresponde a uma determinada informação deixa de ser utilizado e as informações necessárias para que ocorra o intercâmbio entre empresas passam a ser disponibilizados através do DTD, que define não apenas a estrutura

do documento, mas também descreve como eles podem ou não serem usados, o que pode ser colocado em seus interiores e se são ou não elementos obrigatórios do documento.

# CONCLUSÃO

Este estudo abordou aspectos relevantes da tecnologia XML de modo a descrever suas características e benefícios na transferência de informações entre diferentes bancos de dados. As limitações do processo EDI, utilizado até então por empresas que necessitavam trocar informações com seus parceiros comerciais, e o suporte a esta tecnologia por parte dos fornecedores de SGBD têm contribuído para a crescente adesão da XML nas empresas.

De acordo com o objetivo de conhecer melhor a tecnologia XML, o capítulo 1 deste estudo abordou as principais características e as vantagens que estão levando empresas e profissionais de desenvolvimento de sistemas a estudar e reconhecer a XML como um padrão para a manipulação, estruturação e transporte de dados em diferentes aplicações e áreas de negócio.

Conforme descrito no capítulo 2 deste estudo, durante vários anos as organizações que necessitavam trocar informações com seus parceiros comerciais por meio eletrônico utilizaram o processo de EDI. Porém, com o passar dos anos e o aumento da busca por soluções que representassem maior economia de tempo e dinheiro, a implementação complexa, a falta de flexibilidade e o alto custo mostraram-se as maiores limitações do EDI.

Dentro do processo de transferência de documentos, a XML disponibiliza uma plataforma comum de comunicação entre parceiros que utilizem diferentes sistemas e aplicações, além de oferecer maior simplicidade e flexibilidade se comparada ao EDI. Desta forma, a XML mostra-se como uma forte tendência a ser adotada em substituição ao Intercâmbio Eletrônico de Dados, até então comumente utilizado.

Como trabalhos futuros, propõem-se novos estudos que possam se aprofundar nas características do XML, aqui descritas de forma sucinta, possibilitando maior entendimento do universo de conceitos e acessórios relacionados. Além disso, são relevantes estudos para a implementação de um portal B2B (*Business to Business*) utilizando a tecnologia XML na empresa apresentada no estudo de caso.

# REFERÊNCIAS


CHANG, B.; SCARDINA, M.; KARUN, K.; KIRITZOV, S.; MACKY, I.; NOVOSELSKY, A.; RAMAKRISHNAN, N. ***Oracle XML – O Manual Oficial***. Rio de Janeiro: Campus, 2001.

PITTS-MOULTIS, N.; KIRK, C. *XML Black Book*. São Paulo: Makron Books, 2000.

SILBERSCHATZ, A.; KORTH, H.; SUDARSCHAN, S. ***Sistema de Banco de Dados***. 3. ed. São Paulo: Makron Books, 1999.

KROENKE, D. ***Banco de Dados***. 6. ed. Rio de Janeiro: LTC, 1999.

SANTOS, L. C. P. ***Utilizando XML para Publicação de Dados Relacionais***. 2001. 106 f. Tese de Mestrado em Ciência da Computação – Universidade Federal de Minas Gerais, Belo Horizonte, 2001.

MARQUES, P. F. J. V. ***Troca de Informação de Negócio para Negócio – Do EDI ao XML/EDI e ebXML***. 2003. 136 f. Monografia de Engenharia da Computação – Universidade Fernando Pessoa, Porto, 2003.

CESLINSKI, R. A. ***Integração entre XML e Banco de Dados***. Disponível na Internet: <http://www.presidentekennedy.br/publicacoes/I_encontro/Artigos/artigo02.pdf> . Acesso em 20/09/2004.

WORD WIDE WEB CONSORTIUM. ***Extensible Markup Language (XML) 1.0***. 2004. Disponível na Internet: <www.w3.org>. Acesso em: 15/10/2004.

EAN BRASIL. Introdução ao EDI. Disponível em <www.eanbrasil.org.br> Acesso em: 05/09/2004.

TUBINO, D. F. ***Sistemas de Produção***. Porto Alegre: Brookman, 1999.

***EDI***. Disponível na Internet: <www.hostgold.com.br/hospedagem/EDI>. Acesso em 15/09/2004.

CRN BRASIL. ***Mercado Adota o XML***. 2000. Disponível na Internet: <www.crn.com.br/noticias/artigo.asp?id=8669>. Acesso em: 15/09/2004.

CODIPOR. ***XML***. Disponível na Internet: <www.codipor.pt/xml/xml.htm> . Acesso em 10/09/2004.

ALMEIDA, M. B. ***Uma Introdução ao XML, sua utilização na Internet e alguns conceitos complementares***. Disponível na Internet: <www.ibict.br/cienciadainformacao/include>. Acesso em: 14/09/2004.

MOURA, D. F. C. ***XML – Extensible Markup Language***. Disponível na Internet: <www.gta.ufrj.br/~mdavid/xml.htm>. Acesso em: 05/09/2004.

JEVEAUX, P. C. M. ***Introdução ao XML***. Disponível na Internet: <http://www.portaljava.com/home/modules.php?name=Content&pa=showpage&pid=25>. Acesso em 30/09/2004.

SQL MAGAZINE. ***Introdução ao SQL Server***. Disponível na Internet: <http://www.sqlmagazine.com.br/Artigos/sqlserver/06_intro_sqlserver.asp>. Acesso em: 08/11/2004.

MICROSOFT. ***Visão Geral do SQL Server 2000***. Disponível na Internet: <http://www.microsoft.com/brasil/sql/overview/default.mspx> . Acesso em: 08/11/2004.

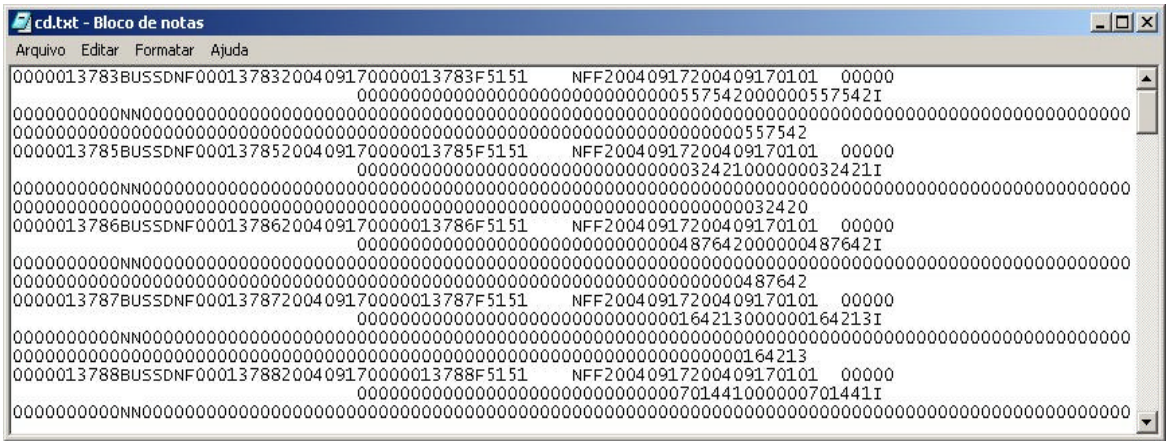
CASTRO, R. G. P. ***Suporte Nativo a XML no SGBD Oracle 9i***. 2002. 46 f. Trabalho de Graduação – Universidade Federal de Pernambuco, Recife, 2002.

***Oracle 9i Database Server***. Disponível na Internet: < http://www.cc6532.hpg.ig.com.br/Sint/Trabalho%20de%20Oracle.doc >. Acesso em 04/11/2004.

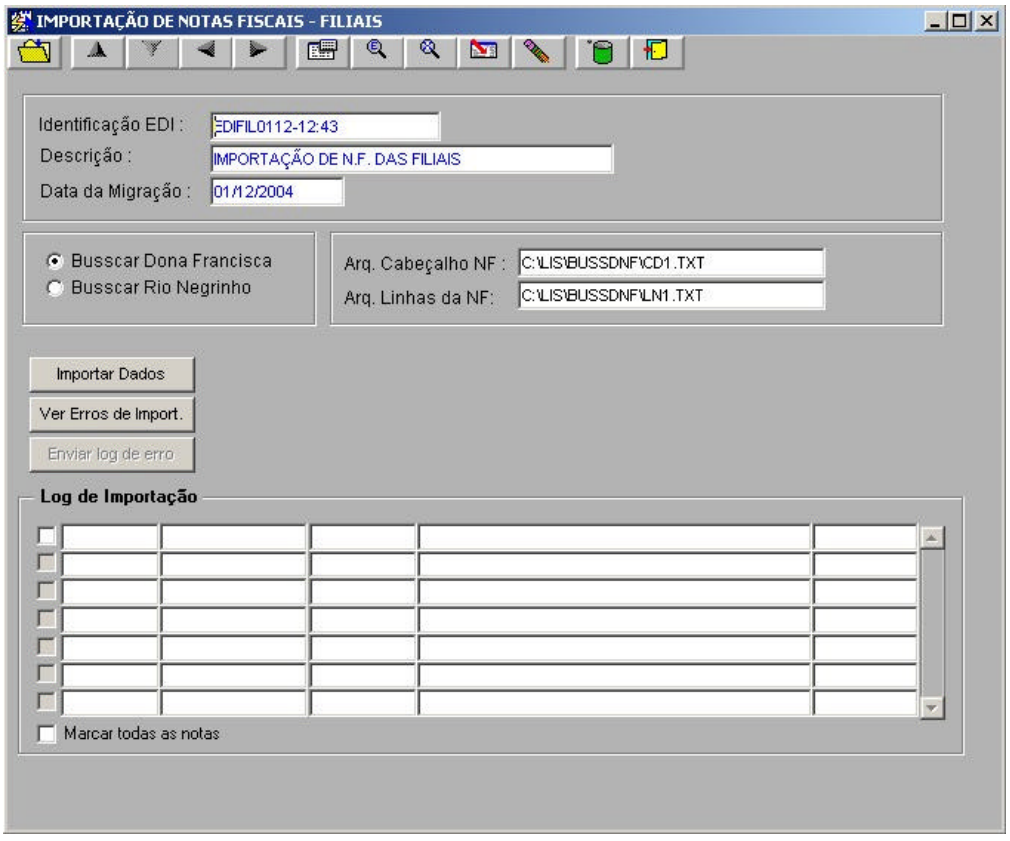
PERFECT XML. ***Exporting SQL Data as XML***. Disponível na Internet: <http://www.perfectxml.com/articles/XML/ExportSQLXML.asp> . Acesso em: 04/11/2004.

# ANEXOS

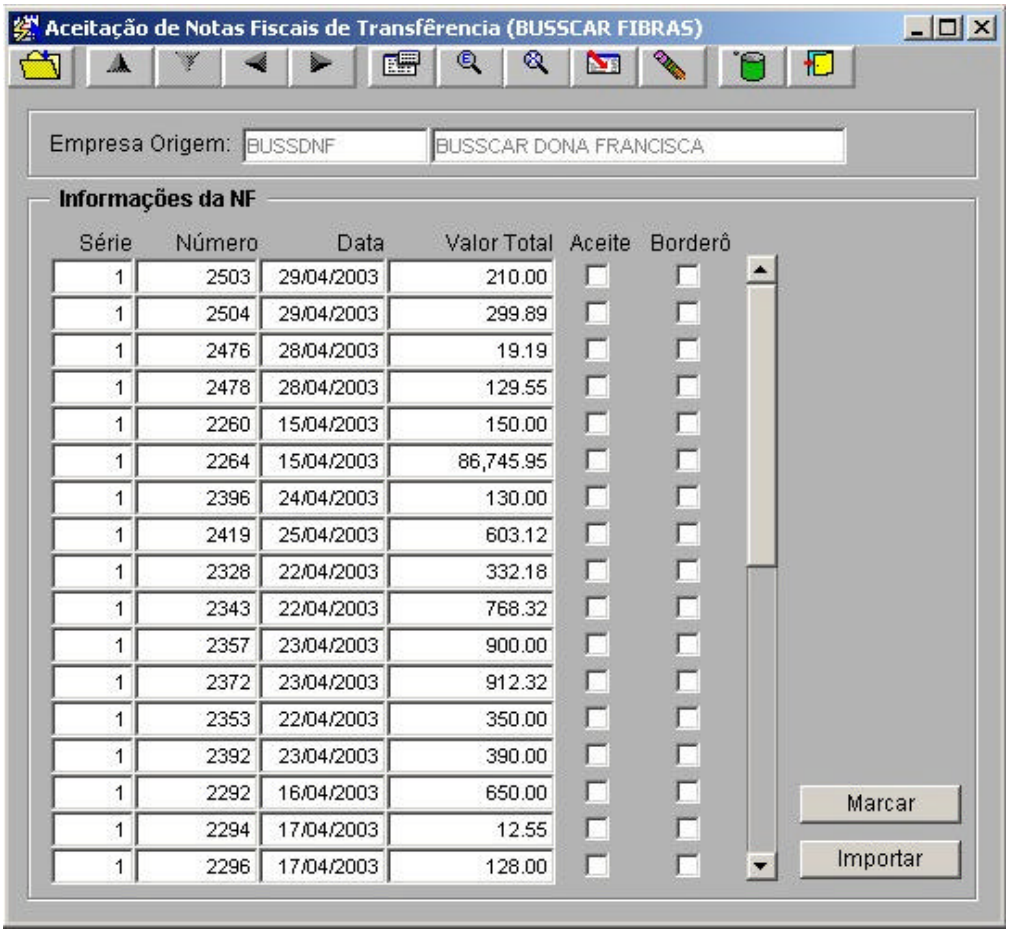
Anexo 01:




Anexo 02:



Anexo 03:


Aceitação de Notas Fiscais de Transfêrencia (BUSSCAR FIBRAS)
Empresa Origem: BUSSDNF BUSSCAR DONA FRANCISCA
Informações da NF
Série Número Data Valor Total Aceite Borderô
1 2503 29/04/2003 210.00
1 2504 29/04/2003 299.89
1 2476 28/04/2003 19.19
1 2478 28/04/2003 129.55
1 2260 15/04/2003 150.00
1 2264 15/04/2003 86,745.95
1 2396 24/04/2003 130.00
1 2419 25/04/2003 603.12
1 2328 22/04/2003 332.18
1 2343 22/04/2003 768.32
1 2357 23/04/2003 900.00
1 2372 23/04/2003 912.32
1 2353 22/04/2003 350.00
1 2392 23/04/2003 390.00
1 2292 16/04/2003 650.00
1 2294 17/04/2003 12.55
1 2296 17/04/2003 128.00
Marcar
Importar