UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Desenvolvimento de um esquema XML para banco de dados sobre ovinos

por

JOÃO ABELAR MARTINS COSTA

Dissertação submetida à avaliação, como requisito parcial para a obtenção do grau de mestre em Ciência da Computação

Prof. Dr. Antonio Carlos da Rocha Costa
Orientador

Porto Alegre, março de 2002.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Costa, João Abelar Martins.
Desenvolvimento de um esquema XML para banco de dados sobre ovinos/ por João Abelar Martins Costa. – Porto Alegre: PPGC da UFRGS, 2002.
88p.:il.
Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2002. Orientador: Costa, Antonio Carlos da Rocha.
1. XML. 2. Esquema XML. I. Costa, Antonio Carlos da Rocha. II. Título.

Sumário

Lista de Abreviaturas

ASCII	American Standard Code for Information Interchange, ou Código americano padrão para intercâmbio de informação
BMP	Basic Multilingual Plane, ou nível multilíngue básico
CDF	Channel Definition Format, ou formato de definição de canal
CSS	Cascading Style Sheet, ou folha de estilo em cascata
DTD	Document Type Definitions, ou Definição de tipos de Documentos
DSSSL	Document Style and Semantics Specification Language, ou linguagem de estilo e de especificação semântica de documento
HTML	Hypertext Markup Language , ou Linguagem de Marcação de Hipertextos
RMD	Declaração de Marcação Requerida
RTF	Rich Text Format, ou formato de arquivo com recursos intercambiáveis
SGML	Standart Generalized Markup Language, ou Linguagem de Marcação Padrão Generalizada
UCS	Universal Character Set, ou conjunto de caracteres universais
XML	eXtensible Markup Languagem , ou Linguagem de Marcação Estendida
XSL	Extensible Style Language, ou linguagem de estilo extensível
W3C	World Wide Web Consortium
WWW	World Wide Web

Lista de Figuras

Resumo

Este trabalho utilizou tecnologias tais como XML (eXtensible Markup Language) e esquemas XML, com objetivo de aprimorar a ovinocultura tornando o setor primário mais competitivo. Foram elaborados arquivos XML com a mesma estrutura (equivalentes) dos arquivos primitivos da Associação Brasileira de Criadores de Ovinos (A.R.C.O.), para que os mesmos possam ser disponibilizados na Internet. Para obter a integridade destes dados na Internet criou-se os esquemas XML, que são arquivos contendo as regras de formação dos dados. Os arquivos XML ficarão protegidos contra dados indesejáveis e disponíveis ao produtor rural via Internet.

Palavras-chave: XML e Esquema XML

TITLE: "DEVELOPMENT OF AN XML SCHEME FOR A DATA BANK CONCERNING SHEEP."

Abstract

This paper has dealt with technologies such as XML (eXtensible Markup Language) and XML schemas, aiming to improve sheep-raising in order to lead the primary sector to a more competitive stance. XML files were created with the same structure (as equivalent) as the original archives of Associação Brasileira de Criadores de Ovinos (A.R.C.O.), the Brazilian Association of Sheep-Breeders, to make them available on the Internet. To secure integrity of such data in Internet, XML schemas were created. The above files are ones which contain rules for data formation. The XML files are protected against unwanted data; and they are available to breeders via Internet.

Keywords: XML and XML Schemas.

1 Introdução

O setor primário, em particular a ovinocultura, vem enfrentando grandes dificuldades, tais como: falta de incentivos fiscais, desvalorização do ovino, dificuldade de comunicação entre a Associação Brasileira de Criadores de Ovinos (A.R.C.O.) e o produtor rural, dentre outras.

Com o auxílio da Internet e fazendo uso de tecnologias tais como XML (eXtensible Markup Language), torna-se possível disponibilizar informações (banco de dados da A.R.C.O.) a um baixo custo para o produtor, desta forma aproximando a A.R.C.O. do meio rural.

O produtor rural terá acesso ao banco de dados sem precisar deslocar-se até a entidade, simplesmente utilizando a Internet como um meio de comunicação.

Estes dados (banco de dados da A.R.C.O.) estarão disponíveis na Internet no formato XML, sendo os mesmos protegidos por um esquema XML, o qual tem por função manter a integridade dos dados do arquivo XML. O esquema XML foi elaborado de acordo com as regras que regem os arquivos de dados originais da entidade (A.R.C.O.), desta forma não permitindo que o produtor cadastre dados indesejáveis.

A A.R.C.O. tem por função disponibilizar os arquivos XML na Internet, bem como capturá-los para efetuar a atualização do seu banco de dados primitivo (após algumas validações dos novos dados). Estes arquivos XML serão atualizados na Internet por parte da A.R.C.O. com uma certa periodicidade, mantendo-os atualizados.

O produtor rural terá acesso aos arquivos XML (via Internet), podendo o mesmo através de softwares especializados fazer a inclusão de novos dados a este arquivo, visto que este encontra-se protegido por um esquema XML, o que garante à A.R.C.O. a integridade dos novos dados.

1.1 Estrutura do Trabalho

O presente trabalho está dividido da seguinte forma: o capítulo 2 apresenta os sistemas de informação na Web. O desenvolvimento da tecnologia XML é apresentado no capítulo 3. O capítulo 4 aborda sobre esquema XML, como sendo um documento controlador, enfatizando suas características .

O funcionamento do sistema de informações da entidade A.R.C.O., de onde foram retirados subsídios para o desenvolvimento deste trabalho, encontra-se no capítulo 5. O capítulo 6 apresenta o banco de dados XML e o esquema XML propostos para permitir a colocação do sistema de informações da A.R.C.O. na Web.

Para finalizar, é apresentado no capítulo 7 uma conclusão sobre este trabalho.

2 Sistemas de Informações na Web

2.1 Métodos de acesso a informações na Web

Os métodos de enviar informações de qualquer forma para um cliente podem ser amplamente separados por categorias em duas abordagens – Push e Pull.

A compra de um jornal e a visita a uma biblioteca são exemplos de métodos de publicação pull. O consumidor tem de fazer um esforço consciente para pesquisar e encontrar informações interessantes. Encontrando algo interessante, digamos , na Segunda-feira, temos que retornar na Terça se quisermos ver as atualizações desse dia. Regularmente temos que iniciar a transferência de informações da sua extremidade da cadeia. Temos de puxá-la (pull) por conta própria.

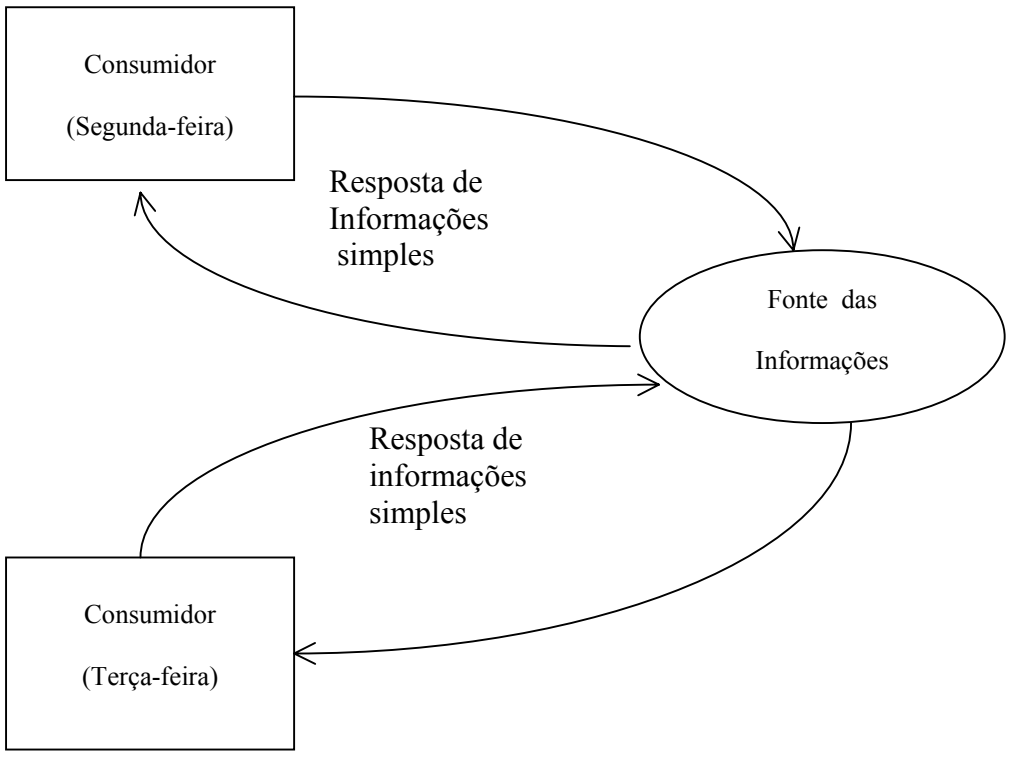


FIGURA 1 - Funcionamento da tecnologia de publicação pull

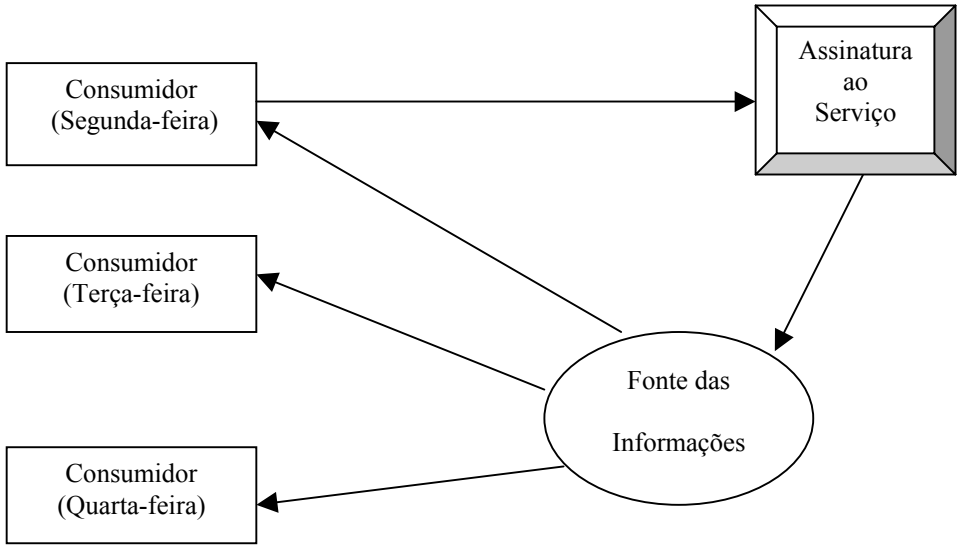


FIGURA 2 - Funcionamento da tecnologia de publicação push

Ligar a TV no noticiário e ouvir os resultados de esportes no rádio são exemplos de publicação "push". Neste modelo, o editor constantemente divulga as informações mais recentes. O usuário tem apenas que ligar a TV ou o rádio para receber automaticamente as novas informações à medida que elas se tornam disponíveis.

A World Wide Web é basicamente um sistema de publicação pull. Se o conteúdo do seu site da Web favorito mudar por alguma razão, não ficará sabendo da mudança até que visite novamente esse site.

Para a maioria das aplicações possíveis da Web, especialmente as aplicações comerciais que requerem o envio imediato de informações, as quais dependem de negócios, o modelo push possui muitas vantagens em relação ao modelo pull. Os usuários, após localizarem informações interessantes de uso para seus negócios, provavelmente irão preferir que as alterações feitas àquela informação apareçam automaticamente ao invés de ter que visitar novamente o site e fazer download da informação. Há muitos exemplos, variando de informações quanto ao preço de estoque, aos anúncios de estatísticas econômicas e aos resultados do futebol. Em um ambiente da intranet, push pode ser usado para manter os funcionários atualizados, quanto a tudo desde os procedimentos de qualidade até aos anúncios da empresa e às tabelas da liga de futebol.

É claro que um mecanismo é necessário para capturar detalhes sobre determinadas fontes de informações (canais). Para cada canal, temos que saber quais as páginas da Web que constituem o conteúdo desse canal. Também temos que saber como verificar as atualizações no canal. Devemos verificar o site a cada hora, todos os dias ou todas as semanas, além disso, armazenar alguns títulos simples e trechos da informação, de modo que podemos fazer download de um documento pequeno que age como índice, listando as informações disponíveis no canal.

Para obter uma tecnologia push, precisamos de uma representação de dados aberta. Precisamos definir as partes do componente da representação, cada qual definirá

um ou mais canais. Cada canal, por sua vez, terá um título e uma programação atualizada associada. Uma programação atualizada terá horas de início e fim opcionais e uma configuração de intervalo obrigatório e assim por diante.

Nas melhores situações, esses documentos seriam facilmente lidos, criados, verificados quanto à sua totalidade e processados.

```
<CHANNEL>
   <TITLE>My truly wonderful news channel</TITLE>
   <LOGO HREF = "www.acme.com/logo.gif"/ >
   <ABSTRACT>Dramatic savings on tape backup units</ABSTRACT>
   <SCHEDULE>
      <INTERVALTIME DAY = "1" />
   </SCHEDULE>
</CHANNEL>
```

Se as ferramentas de navegação pudessem estar cientes de tal formato de definição de canal padronizado, elas poderiam usar esses documentos para permitir que os usuários naveguem/assinem canais interessantes a partir do conforto do seu ambiente de navegação na Web familiar. Esses documentos de canal podem ser usados como "cartões de assinatura" pelo navegador para permitir que ele procure, de modo inteligente e automático, e faça download de atualizações do conteúdo.

Além de alguma simplificação e aprumo, o trecho de programa anterior está de acordo com a iniciativa de aplicação da XML conhecida como CDF (formato de definição de canal). O processo de assinar e, subsequentemente , receber atualizações de um canal é ilustrado a seguir [HAP 99].

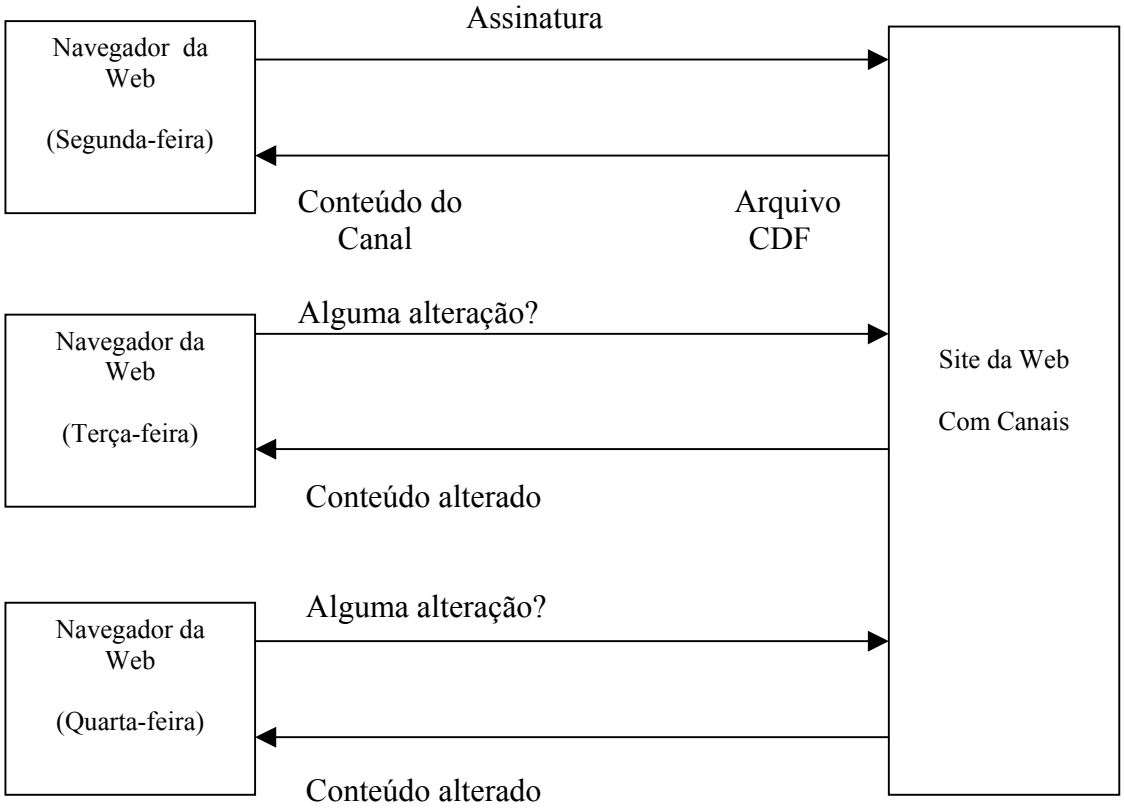


FIGURA 3 - Assinando e recebendo atualizações de um canal

A XML possui recursos para que se possa criar documentos com a tecnologia (Push), onde os dados serão atualizados de forma automática, conforme a necessidade do usuário, sem que este precise fazer um download para saber se novas informações estão disponíveis.

Devido aos recursos disponibilizados pela XML e ao crescimento do comércio eletrônico, podemos dizer que a XML em um futuro muito próximo irá modificar a forma de comércio utilizada atualmente.

2.2 Banco de Dados na Web

Muitas informações no mundo estão armazenadas em bancos de dados, terabytes de arquivos de dados pessoais, registros de saúde, resultados de partidas de futebol, preços de mercadorias, e assim sucessivamente. A maior parte dessas informações pode ser colocada em uso na Web. O problema está em como chegar a elas. Uma conversão direta em uma página de Web não é uma opção porque esses bancos de dados precisam ser constantemente atualizados e , no geral, se alteram rapidamente. O que é necessário é um mecanismo que permita que tais bancos de dados sejam acessados/modificados através de uma interface da Web, e ainda assim se mantenham em seus formatos originais de banco de dados.

Uma das grandes atrações da web é que os usuários não precisam saber ou se importar em como as páginas da Web são criadas pelos servidores da Web. Um navegador da Web simplesmente emite uma solicitação e aguarda a resposta. Em muitos casos, as páginas da Web são estáticas, permanecendo no servidor aguardando serem solicitadas, conforme ilustrado abaixo.

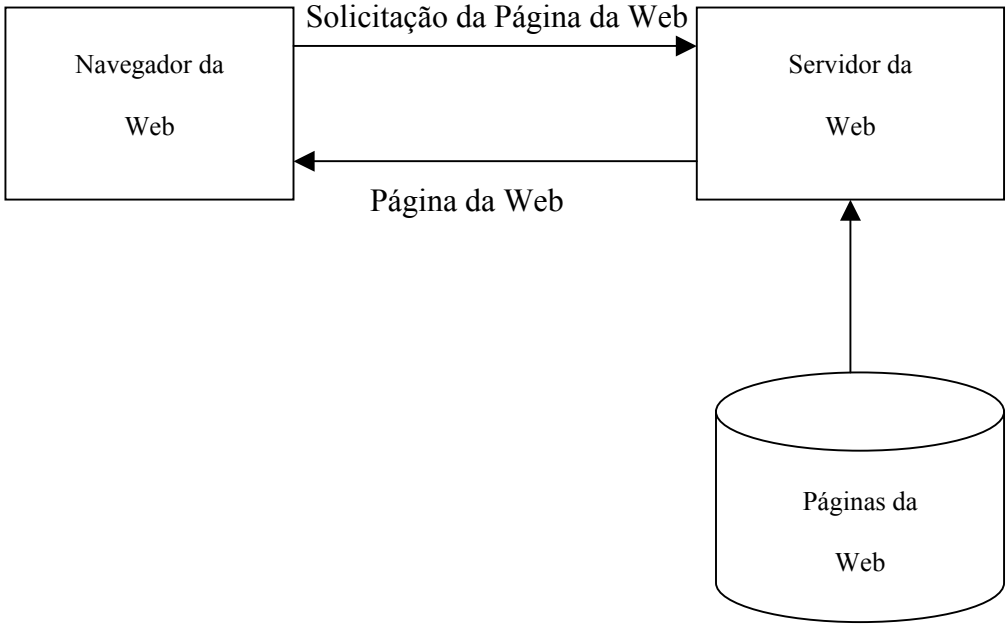


FIGURA 4 - Páginas da Web estáticas servidas por um site da Web

Entretanto, pode-se introduzir um nível de procedimento indireto nesse local, completamente transparente para o navegador da Web, conforme abaixo.

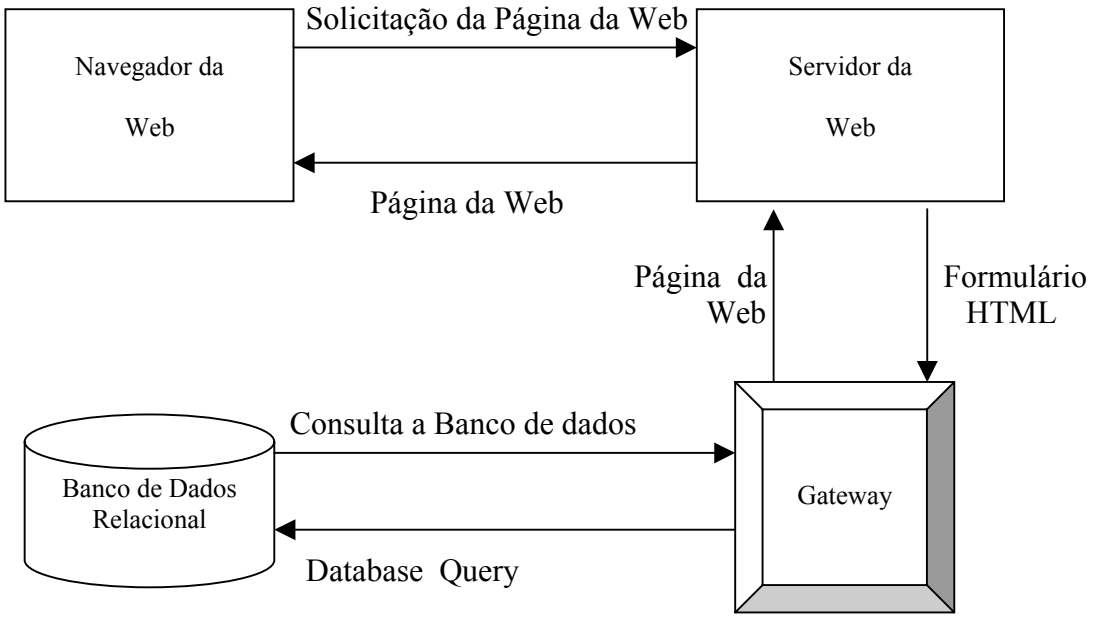


FIGURA 5 - Páginas da Web criadas dinamicamente, servidas por um site da Web

O que é necessário é uma maneira de se capturar informações relacionadas a banco de dados, tais como tabelas, campos, valores e outros, em uma forma que possa ser traduzida para HTML no mesmo instante, para que tudo pareça transparente para os aplicativos que lidam somente com HTML diretamente, ou seja, o navegador da Web.

Embora possamos argumentar que a XML pode fazer tudo que um banco de dados tradicional também pode, há necessidade de se construir pontes a tecnologia de banco de dados existentes, especialmente a tecnologia de banco de dados relacional. Da forma com que a Web funciona, ela proporciona a si mesma uma integração uniforme de ambas as partes através de conversões imediatas, como ilustra o exemplo.

Historicamente, essas conversões tem sido para a HTML, mas como as ferramentas de navegação XML nativas se tornam a tendência principal, a necessidade de se realizar uma conversão para HTML pode ser removida, como ilustrado abaixo.

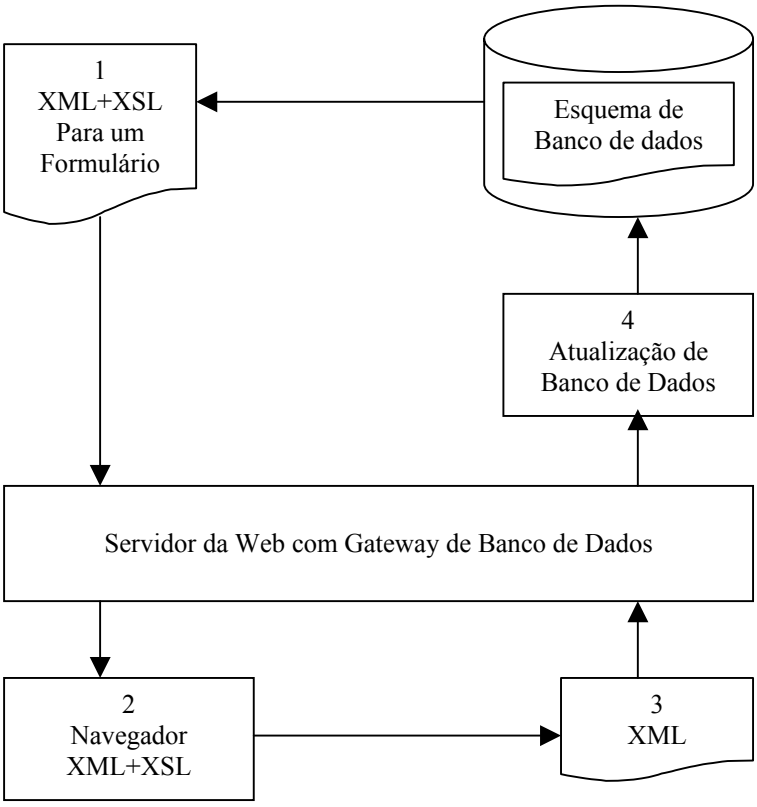


FIGURA 6 - XML+XSL nativas usadas para realizar uma atualização a um banco de dados

As vantagens de não usar a conversão em HTML podem ser significativas, especialmente no caso onde os resultados de banco de dados estejam sendo enviados de volta para o navegador da Web para exibição.

Devido a maior parte dos dados encontrarem-se em banco de dados, surgiu a necessidade de disponibilizar estes dados na Web.

Estes dados (tabelas, campos, valores e outros) podem ser disponibilizados usando a HTML, para isto é necessário capturá-los e traduzi-los no mesmo instante (para HTML), apresentando os resultados ao usuário de uma forma transparente.

A XML possui recursos para lidar diretamente com estes dados (banco de dados relacional) através de conversões imediatas.

Para que possamos trabalhar com comércio eletrônico é de vital importância o tratamento com banco de dados, pois nestes concentram-se as principais informações.

3 XML

XML (Extensible Markup Language) é um sistema de codificação que permite que qualquer tipo de informação seja distribuido através da WWW. Ao contrário do HTML, a XML é verdadeiramente para todos os propósitos. A XML oferece o panorama de uma ampla variedade de aplicações, cada uma servindo a uma função em particular e usando a Web como um mecanismo de distribuição.

A XML é uma evolução do HTML, não apenas convive, como se integra facilmente com ele. O IE 5.0 (Internet Explorer 5.0) foi o primeiro produto a adotar a novidade. O browser da Netscape, que chegará ao mercado também abraçará a XML. A XML e o HTML têm uma origem em comum, o SGML, como visto anteriormente.

XML é uma metalinguagem definida como um subconjunto de SGML, cujo objetivo é fornecer os benefícios não existentes em HTML e ser mais fácil de utilizar do que SGML.

Em XML, projetistas podem criar seus próprios elementos de acordo com a aplicação que está sendo modelada, dando importância ao conteúdo e à estrutura da informação, sem se preocupar com a apresentação. Para que o parser XML verifique se um documento está correto ou não (parser de validação), ele processa inicialmente seu DTD correspondente, para verificar a estrutura do documento [BRD 97]. Vários parsers, disponíveis gratuitamente no WWW, verificam (análise léxica) e validam (análise sintática) documentos XML de acordo com seu DTD associado. Além desses, há também parsers que não exigem a presença de um DTD e que portanto somente validam seus elementos.

Outros benefícios disponíveis no WWW são os editores de documentos XML. Eles permitem a criação simplificada de documentos XML (com ou sem o DTD) possibilitando inserir e eliminar elementos, propriedades e informações associadas, mudar parte de estrutura para outro local e validar o documento. Exemplos de editores para documentos XML são: Microsoft XML Notepad, Xeena e XED.

Pode-se notar que o documento XML não trata a apresentação das informações, mas somente o conteúdo a ser apresentado. Sendo assim, existe a necessidade da utilização de outro recurso que é responsável pelos atributos de apresentação. Esta tarefa pode ser realizada através da utilização de parsers específicos ou linguagens apropriadas para associar estilos ao conteúdo de um documento XML.

3.1 O Padrão SGML e a XML

A SGML (Standard Generalized Markup Language) ou Linguagem Padrão de Marcação Generalizada é um exemplo de um sistema de marcação generalizada, foi publicado pela primeira vez em 1986. A SGML oferece um esquema de marcação simples, independente de plataforma e extremamente flexível. Ela é simplesmente uma maneira de representar documentos, uma metalinguagem (linguagem para definir linguagens de marcação).

A SGML não é um software. Se quiser fazer algo prático com um documento que foi codificado em SGML, você precisará de um software que entenda SGML e realize a função desejada. Por causa do tamanho e complexidade do padrão SGML inteiro, tal software tende a ser relativamente caro.

Até 1996, a comunidade SGML como um todo não havia tido um interesse efetivo pelos problemas da Web. As aplicações de ajuda que funcionavam com os browsers Web tinham sido desenvolvidas para permitir que documentos SGML fossem distribuídos na Web e visitados pelos clientes. Parecia que somente duas escolhas estariam disponíveis aos provedores de informações para um público minoritário que importará o suficiente para adquirir uma aplicação de ajuda especializada, ou adotar o universalmente aceito no meio HTML, com suas limitações.

Na metade de 1996, um grupo de aproximadamente 80 peritos juntaram forças com o World Wide Web Consortium (W3C) para formar um grupo de trabalho sob a chefia de Jon Bosak da Sun Microsystems. A meta do grupo era desenvolver uma linguagem de marcação que tivesse o poder e a generalidade da SGML e, ao mesmo tempo, fosse fácil de ser implementada na Web. Essa linguagem de marcação teria de fazer o seguinte:

- Dar suporte à marcação generalizada na Web;
- Produzir documentos que idealmente fossem válidos de acordo com o livro de regras da SGML;
- Fornecer suporte para hiperlinks que fossem altamente compatíveis com a abordagem URL;
- Fornecer um mecanismo de folha de estilo genérico e poderoso.

A primeira façanha deles foi desenvolver uma especificação inicial de linguagem para a XML, a qual foi divulgada em novembro de 1996 na conferência SGML de 96 em Boston, EUA. Um segundo esboço foi publicado em março de 1997.

Logo após, em abril de 1997, o primeiro esboço da especificação de hiperlinks XML foi publicado.

Em 1º de julho de 1997, essa organização do tipo projeto de trabalho foi formalizada nos padrões das linhas da W3C. O conselho de revisão editorial W3C SGML tornou-se o grupo de trabalho W3C XML (GT) e agora segue as linhas dos grupos de trabalho da W3C. Este GT W3C XML está se encarregando da formalização do padrão XML. Na mesma data, o atual grupo de trabalho SGML transferiu suas funções para o GT XML, que mudou seu nome para grupo de interesse especial W3C XML. Essas mudanças regularizam a posição da XML como uma atividade aprovada pela W3C.

O grupo de trabalho SGML tem a XML como uma realização sua. A XML é, ao mesmo tempo, menos que a SGML e mais que a SGML. Formalmente, a especificação da linguagem XML é um perfil da SGML. Menos formalmente é um subgrupo. O grupo de trabalho SGML tem selecionado somente aqueles recursos da SGML que acha absolutamente necessários para a Web. O grupo jogou o resto do padrão fora.

Então, a XML é muito menos do que a SGML. Ela é menos complexa e menos carregada de todos aqueles recursos engenhosos (muitos deles opcionais) que têm provado serem problemáticos para os programadores que almejam desenvolver software que concorde com a SGML.

Por outro lado, a XML mantém os benefícios-chave que a SGML oferece. Com a XML temos uma marcação generalizada, permitindo inventar seus próprios tagsets (grupos de tags). Permite fazer documentos autodescritivos e com possibilidade de validação.

Os documentos grandes podem ser divididos em partes, para ficarem mais fáceis de administrar, pode-se distribuir documentos parciais ou complexos na Web e assim por diante. A XML classifica-se como uma solução "80/20" 80% dos benefícios da SGML em troca de 20% de sua complexidade.

A XML para ser prática na Web precisa ter capacidade de circulação e estilo já embutida nela. A SGML não tem mecanismo de estilo, embora forneça as ferramentas com as quais um esquema de hiperlinks pode ser construído, ela não define esse mecanismo de vínculo.

Antes de entrar nos detalhes das diferenças entre XML e a SGML devemos sempre ter em mente que a HTML é uma aplicação da SGML e a XML é um subgrupo da SGML.

Essa é uma distinção muito importante, porque, embora quase todas as ferramentas desenvolvidas para uso com a SGML possam ser facilmente usadas com a HTML, poucas ferramentas SGML existentes podem ser usadas para a XML. Por exemplo, um parser (programa) de validação completamente SGML não pode validar documentos XML a não ser que este seja modificado para aceitar a XML.

Como qualquer outra aplicação SGML, a HTML é definida por uma declaração SGML fixa em uma DTD. O sucesso da HTML na World Wide Web pode ser atribuido, pelo menos em parte, ao fato de que o lado SGML da HTML é quase completamente ofuscado, porque quase tudo é diretamente ligado ao navegador da WWW.

A XML também usa uma declaração SGML fixa, e ela pode ter uma DTD SGML. Mas diferente da HTML, a XML impõem certas restrições ao uso da linguagem SGML e essas restrições não podem ser expressas em uma declaração SGML.

A SGML é uma metalinguagem, uma linguagem para definir linguagens (DTDs) que definem as aplicações SGML (geralmente, mas não exclusivamente, a estrutura dos documentos). Através do mecanismo de uma declaração SGML, a SGML também tem o poder de redefinir a si própria.

A XML é também uma metalinguagem, uma linguagem para definir DTDs XML para aplicações SGML. Para que a XML funcione como a HTML, ela deve possuir uma declaração específica para a SGML. Essa declaração define o grupo de caracteres XML, a sintaxe e os recursos SGML que são aceitos ou explicitamente proibidos. É possível mudar a declaração SGML para HTML. O preço é que um navegador Web pode não ser capaz de interpretar nenhum dos seus arquivos HTML, mas um editor SGML não terá nenhum problema com as mudanças. Em contraste à HTML (ou qualquer outra aplicação SGML), é proibido para XML possuir uma explícita declaração SGML.

Como a XML é um subgrupo da SGML, seria natural assumir que a XML é muito menos poderosa que a SGML e que se pode fazer muito menos com ela. Isso é verdade até certo ponto, a XML realmente impõe um bom número de restrições no uso da sintaxe do padrão SGML, mas não necessariamente a custo do desempenho.

A XML não deveria simplesmente ser tida como uma versão inferior da SGML. Além das restrições a alguns dos recursos da SGML, a XML também adiciona alguns poderosos recursos à SGML e pode ser tida como um refinamento da SGML para torná-la mais apta para uso na Internet.

Em sua forma original, o padrão SGML era muito favorável ao uso do alfabeto ocidental "padrão", o que é conhecido como grupo de caracteres ASCII. Em retrospecto, isso foi um erro por parte das pessoas que desenvolveram a SGML. As línguas européias (com caracteres acentuados) poderiam ser acomodadas permitindo o uso de entidades SGML adicionais dos padrões ISO. Porém, embora o uso de

entidades externas permitisse que o conteúdo de instâncias SGML (documentos) usassem caracteres não latinos, ele não permitia que esses caracteres fossem usados para as tags(marcadores) dos próprios elementos (na DTD SGML).

Essa obrigatoriedade acidental do uso da ASCII estava na verdade em conflito com um dos princípios básicos da própria SGML (cláusula 0.2, subclasse e, afirma: " Não haverá nenhuma predisposição quanto ao idioma."). Em dezembro de 1996, um TC chamado TC for Extended Naming Rules (TC para as Regras de Nomeação Estendida) foi publicado para corrigir essa situação.

As correções das Regras de Nomeação Estendida adicionaram um Anexo J ao padrão SGML. Este anexo estende a sintaxe da declaração SGML para permitir a definição de grupos de caracteres estendidos e o uso de grupo de caracteres que não fazem distinção entre caracteres maiúsculos e minúsculos.

O Anexo J é uma extensão opcional da SGML, e um sistema SGML não precisa ser capaz de aceitar as Regras de Nomeação Estendida para estar em conformidade com o sistema SGML. Porém, a implementação das mudanças neste anexo são compulsórias na XML, as mudanças estão incluídas na declaração SGML da XML e devem ser aceitas por todas as ferramentas XML [LIR 97].

A SGML oferece uma plataforma muito complexa, dificultando trabalhar com a mesma, por este motivo foram criados subgrupos da SGML com as mesmas características, porém com menor grau de complexidade, dentre estes subgrupos encontramos a XML e a HTML.

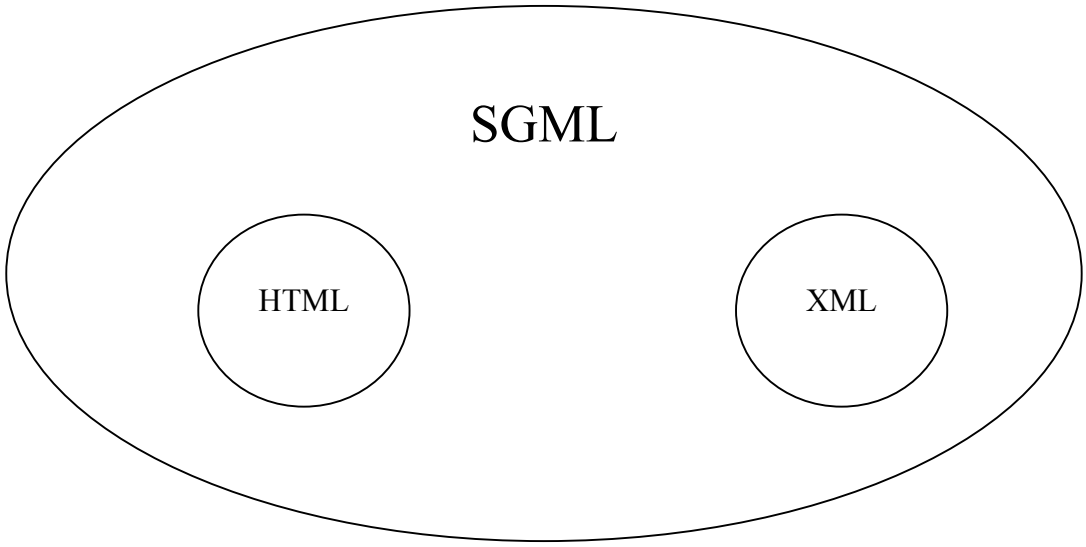


FIGURA 7 - A HTML e a XML fazem parte de um padrão SGML

Vemos que a SGML é a mais completa linguagem de marcação usada para trabalhar na Web.

Devido a SGML ser muito complexa surgiram sub-grupos, tais como a HTML e recentemente a XML. Estes grupos possuem as mesmas características da SGML, com algumas restrições, restrições estas que tem por objetivo torná-las mais simples, devido trabalharem com um número reduzido de comandos.

A SGML continua sendo compatível com os seus sub-grupos, com pequenas restrições no que diz respeito a sintaxe, onde os sub-grupos possuem uma sintaxe mais simples. Por exemplo, considerando um determinado comando, sua sintaxe em XML é reconhecida pela SGML, sendo que este mesmo comando com sua sintaxe completa em SGML pode não ser interpretada pela XML, devido ao nível de recursos que esta disponibiliza ser maior que na XML.

Sendo assim, podemos dizer que a SGML é a origem de tudo e devido sua complexidade e abrangência foi subdividida, gerando recursos menores e menos complexos para fins mais específicos, tais como:

- XML para arquivos de dados;
- XSL para gerar folha de estilos;
- HTML para criar páginas na Web.

3.2 O padrão Unicode e a XML

O crescimento constante dos mercados de software Internacional tem contribuído de uma forma progressiva para a representação de texto em várias línguas. O unicode veio superar as limitações impostas pelas incompatibilidades dos principais standards (da época) de representação de caracteres, e é presentemente um dos sistemas mais usados na codificação de caracteres, que tem vindo progressivamente ao longo dos últimos anos a ser alvo de uma atenção mais cuidada, devido à sua importância como elemento fundamental no processo da internacionalização.

A Internet, bem como a computação em geral, tem uma reputação antiga de dar preferência ao idioma inglês. Tudo, desde as linguagens de programação aos nomes de tipo de elemento HTML, é definido em inglês. O inglês utiliza poucos símbolos. O conjunto (letras, dígitos, pontuação, etc.) requerido encaixa-se nos 128 códigos disponíveis no que são conhecidos como ASCII de 7 bits. A maioria das máquinas utiliza um byte (8 bits) como sua unidade de armazenamento básico, e uma grande proporção do software existente no mundo foi criada com a premissa de que 1 byte comporte 1 caracter.

Infelizmente, caracteres de byte único estão longe de serem adequados para lidar com diversos idiomas e conjuntos de caracteres em uso no mundo, um fato que causou grande alívio proporcionado pela Web, que transformou o mundo em uma gigantesca imprensa virtual. Basta apontar o navegador da Web para a pizzaria do outro lado do mundo, da mesma forma que se pode apontá-lo para a pizzaria na esquina de sua casa. Se estiver utilizando um navegador que espera que os caracteres se encaixem em um único byte, provavelmente verá um rabisco na tela, se tal página estiver em chinês ou árabe, por exemplo.

O padrão Unicode é um modo de lidar com conjuntos de caracteres que resolvem esse problema. O Unicode é um único conjunto de caracteres padronizado que pode ser utilizado para lidar com os idiomas inglês, árabe, chinês, etc. Qualquer sistema compatível com o Unicode pode exibir informações codificadas com o Unicode corretamente, independentemente do idioma no qual foram escritas.

Até há relativamente poucos anos, não era possível visualizar gráficos num PC, a qualidade das impressoras era bastante pobre e nem tão pouco permitiam a impressão de caracteres não-latinos.

A possibilidade de usar um computador para escrever caracteres de países cujos alfabetos contêm caracteres especiais, é um fenômeno relativamente novo.

A evolução tecnológica permitiu um desenvolvimento acentuado em inúmeros países, na sua maior parte da Europa de Leste, Arábia e sobretudo Asiáticos. Alguns destes países, devido à sua linguagem pictórica, tinham extrema dificuldade em transmitir informação de modo digital, tanto entre eles, como com os restantes.

O desenvolvimento de software nestes países, era apenas uma tarefa para engenheiros. A sua complexidade fazia com que só estes os pudessem utilizar, tornando-os assim, inacessíveis à grande maioria da comunidade.

Neste cenário, algumas multinacionais da área da informática, desenvolveram alguns programas multi-linguísticos, nos quais incluíam também alguns países asiáticos. O primeiro e o melhor programa multi-linguístico foi o "Star Viewpoint" desenvolvido pela Xerox. Este, e outros programas desenvolvidos pela IBM, destinavam-se apenas a grandes companhias, ficando assim, (e mais uma vez) inacessíveis à maioria da população.

O aparecimento dos computadores Macintosh, criou grandes expectativas devido ao seu potencial gráfico, condição ideal para o uso de programas multi-linguísticos, mas o sistema operativo, apesar de suportar códigos para países Asiáticos, não suportava programas multi-linguísticos.

O mercado de software continuava a crescer em todo o mundo, e a necessidade de representar texto em várias línguas, acentuou-se ainda mais, mas os obstáculos para escrever este tipo de software eram inúmeros. Entre eles verificavam-se os seguintes:

- Codificação declarativa;
- Codificação de variáveis de tamanho variável;
- Sobrecarga de códigos de caracteres e sistemas de fontes.

Preocupados com as limitações impostas pelas incompatibilidades dos principais standards (da época) de representação de caracteres: ASCII - Língua Inglesa; JIS - Japonês; GB - Chinês da Republica Popular da China; BIG-5 - Chinês de Taiwan; KS - Língua Coreana e ISCII - para as línguas Indianas , em 1980, um grupo de engenheiros e especialistas em linguística, iniciou um conjunto de projetos paralelos que para combater esses obstáculos que mais tarde veio dar origem ao Unicode.

O Unicode tem sido o conjunto de caracteres padrão de todas as especificações W3C desde 1997. Especificamente, é o conjunto de caracteres básico, tanto da HTML versão 4.0, quanto da XML versão 1.0.

3.2.1 Visão geral do Unicode

O Unicode é um conjunto de caracteres de 16 bits que atribui números exclusivos para cerca de 38.000 caracteres diferentes usados nos idiomas do mundo. Dentre os idiomas que podem ser codificados no Unicode incluem o árabe, anglo-saxão, devanagari (da Índia), russo, grego, hebraico, tailandês e sânscrito. Além disso, os idiomas ideográficos como chinês e japonês, são suportados, com cerca de 20.000 caracteres ideográficos definidos por grupos de padrões na China, Japão, Coréia e Taiwan. Dos 65.000 números disponíveis, cerca de 18.000 não estão atribuídos no momento e estão disponíveis para expansão futura.

O Unicode pode ser visto como um super conjunto dos conjuntos de caracteres geralmente utilizados em sistemas de 8 bits, tais como ASCII, ISO –8859-1, dentre outros.

Caractere	nº de caracteres ASCII (Binário)	Caractere	nº de caracteres Unicode (Binário)	nº Unicode caractere (Decimal)
9	0011 1001	9	0000 0000 0011 1001	57
A	0100 0001	A	0000 0000 0100 0001	65

Para permitir que os sistemas que podem lidar somente com conjuntos de caracteres de 7 ou 8 bits lidem com o Unicode, dois formatos de transformação estão definidos no padrão. Usando esses formatos, pode-se converter Unicode de 16 bits em um equivalente de 8 bits retornando de novo a 16 bits sem perda de informação. Esses formatos de transformação são conhecidos como UTF-8 e UTF-7.

- UTF-8 é uma codificação de transformação que utiliza códigos de 8 bits para representar caracteres Unicode. No UTF-8, todos os caracteres US ASCII têm representação normal. Os caracteres Unicode fora dessa faixa são codificados como um número variável de bytes (até 6 bytes por caractere). Quando múltiplos bytes são requeridos, o primeiro byte indica quantos bytes serão necessários.
- UTF-7 é uma codificação de transformação que utiliza códigos de 7 bits para representar Unicode.

Um terceiro formato de transformação, o UTF-16 especifica como o Unicode que pode conter substitutos pode ser convertido de/para ISO 10646.

Muitas das linguagens de programação do mundo foram originadas em 8 bits. As linguagens tais como ANSI C, Pascal, Cobol, Clipper e outras são linguagens de 8 bits. Algumas linguagens como o C++ contêm suporte para Unicode na forma de bibliotecas de “caracteres ampliadas”. Algumas linguagens mais modernas, como o Java, contêm suporte direto para caracteres ampliados. Os sistemas operacionais com suporte direto ao Unicode também estão se tornando cada vez mais comuns. Exemplos incluem Windows NT, AIX e Plan 9.

3.2.2 Uso do Unicode na XML

De todos os processadores compatíveis da XML exige-se que os formatos de transformação UTF-8 e UTF-16 sejam lidos. Se o UTF-16 estiver sendo usado, a marca de ordem de bytes será requerida. Cada entidade em um documento XML pode usar UTF-8 ou UTF-16.

Além disso, um processador da XML pode implementar suporte para outras codificações de caracteres, se quiser. Se outras codificações estiverem em uso, então uma declaração de codificação deverá ser fornecida para cada entidade, isto é:

<?xml encoding = "EUC-JP"?>

Na ausência de uma declaração de codificação ou de uma marca de ordem de bytes, um processador da XML partirá do pressuposto que os dados estão no formato UTF-8.

Observe que, independentemente da codificação especificada na declaração de codificação, quaisquer referência a caracteres como "8#x0041;" sempre farão referência ao conjunto de caracteres Unicode.

Na falta de instruções ao contrário, um processador da XML parte do pressuposto que uma entidade foi codificada em UTF-8. Caso possua um conjunto de arquivos US ASCII, eles serão automaticamente compatíveis com a XML porque são considerados como estando no formato de transformação UTF-8. Esse fato requer alguma explicação.

O UTF-8 foi criado na Bell Labs por um grupo que incluía os projetistas dos sistemas operacionais Unix e Plan 9. A idéia era criar uma codificação de caractere que permitisse caracteres amplos, mas que não rompessem a grande base de sistemas e protocolos existentes que espera dados de 8 bits.

O UTF-8 é uma codificação de multibytes do Unicode. Em outras palavras, alguns caracteres têm 1 byte, outros têm 3 bytes e assim sucessivamente . Até 6 bytes são utilizados para representar um único caractere. Se mais de 1 byte estiver sendo utilizado, parte do primeiro byte indicará quantos bytes serão necessários. Caracteres de byte único correspondem exatamente a ASCII de 7 bits [HAP 99].

Todas as linguagens de programação precisam trabalhar com códigos para que possam representar informações.

O padrão mais utilizado é o ASCII, sendo que este possui suas restrições, tais como um número reduzido de caracteres.

A maioria das linguagens utiliza 1(um) byte para representar estes códigos, sendo 1 (um) byte formado por 8 (oito) bits e cada bit com possibilidade de ter 2 (dois) estados possíveis (0,1), então temos $2^8 = 256$ possibilidades diferentes. Este é um número muito pequeno para que se possa fazer a representação de todos os códigos dos vários idiomas existentes, por este motivo foi criado o Unicode, que é um padrão que não segue as regras citadas acima , englobando uma enorme quantidade de códigos diferentes, possibilitando lidar com idiomas tais como inglês, árabe, chinês, etc.

O Unicode é um conjunto de 16 bits, permitindo atribuir números exclusivos para cerca de 38.000 caracteres diferentes usados nos idiomas do mundo, o que seria praticamente impossível fazer com 1 (um) byte onde temos somente 256 possibilidades diferentes de códigos distintos.

Devido a maioria dos softwares trabalharem com 1 (um) byte foram criados dois formatos de transformação conhecidos como UTF-7 e UTF-8, que tem por objetivo

converter Unicode de 16 bits em um equivalente de 8 bits retornando de novo a 16 bits sem perda de informação.

Com o uso do Unicode é possível globalizar as informações, sem restrições quanto ao idioma.

3. 3 Relação existente entre a XML e a HTML

A XML é uma nova grande maneira de marcar informações na Web, não significando desta maneira o fim da HTML.

A HTML e a XML operam em níveis diferentes de generalidade, não entrando em uma competição direta. A HTML é uma aplicação da SGML, o que significa que ela fornece um grupo específico de tipos de elementos, com um objetivo particular, exibir páginas on-line na Web, com hiperlinks. Por outro lado, a XML é um perfil SGML, o que significa que pode dar suporte a um intervalo ilimitado de aplicações. Algumas dessas aplicações podem ser do tipo HTML em uma esfera, mas a maioria terá objetivos e projetos muitos diferentes.

A XML será mais interessante para pessoas e organizações que têm recursos de informações que não se encaixam no molde da HTML e recursos que eles queiram deixar disponíveis na Web.

Estes são alguns exemplos:

- Livros;
- Transações financeiras;
- Manuais técnicos;
- Fórmulas químicas;
- Registros médicos;
- Catálogos de registros para museus;
- Jogos de xadrez .

O papel da XML será maximizado em situações em que os recursos de informação são de longo prazo (já que documentos XML válidos concordam com o padrão internacional – SGML), há também relações complexas neles e entre eles (já que as vantagens de hiperlink da XML permitem que essas relações sejam expressas de uma maneira que é independente de sistema), ou se eles devem ser postos em diferentes usos (porque é muito mais fácil mudar o objetivo da informação onde ela está marcada de uma maneira generalizada). Nestas circunstâncias, as pessoas estarão dispostas a fazer um trabalho adicional para montar uma aplicação XML.

Uma possibilidade interessante é que as aplicações XML podem ser usadas para melhorar as aplicações HTML. Por exemplo, a proposta, Meta Content Framework, da Netscape , envolve o uso de documentos codificados em XML para descrever e indexar sites Web. A XML contém a descrição de cada página e um hiperlink para a página real.

```
<page id=http://www.acc.com/scorpions.html>
<description>Scorpions in the Sun</description>
</page>
```

Um link inverso, da página HTML para a descrição de página, pode ser criado usando-se o elemento link da HTML.

3.4 Localização de Informações (Tradução de Software)

A Web acelerou a velocidade na qual o planeta está encolhendo. A velocidade com que as informações digitais podem ser publicadas para uma audiência mundial torna-se extremamente importante obter-se versões localizadas (traduzidas para outro idioma) do software/documentação disponíveis após o lançamento da, digamos, versão do idioma em inglês.

O processo de localização é lento e, sendo assim, muito caro pela diversidade de representações de dados na qual o material a ser traduzido pode existir. Dezenas de linguagens de programação, formatos de arquivos gráficos, formatos de arquivos de recursos, dentre outros, estão disponíveis. Muitos desses formatos são incompatíveis entre si. Com cada nova representação de dados ou variação de uma representação existente, os pobres tradutores se deparam com mais problemas do que soluções.

Muitos dos problemas da tradução entre os idiomas originam-se do fato que os formatos de documentos tradicionais combinam formatação e conteúdo, de forma que ficam bem próximos entre si . Frequentemente, a única forma de se chegar ao texto é usando a mesma ferramenta de edição que foi usada para criá-lo originalmente. Essa restrição está longe do ideal do ponto de vista da tradução.

Se, no que diz respeito à representação de dados, fosse combinado que o conteúdo ficasse separado, temporariamente, da formatação, mantendo o vínculo entre os dois, a vida seria bem mais fácil para os tradutores. Eles poderiam traduzir o conteúdo de parte de um documento sem ter de usar o pacote de software usado para criá-lo. O software poderia então mesclar a formatação original novamente com o conteúdo para recriar o documento original. Além disso, se a representação de dados referente ao conteúdo pudesse ser padronizada, uma variedade de ferramentas de fornecedores diferentes poderiam operar nos mesmos dados. Esse processo é ilustrado a seguir.

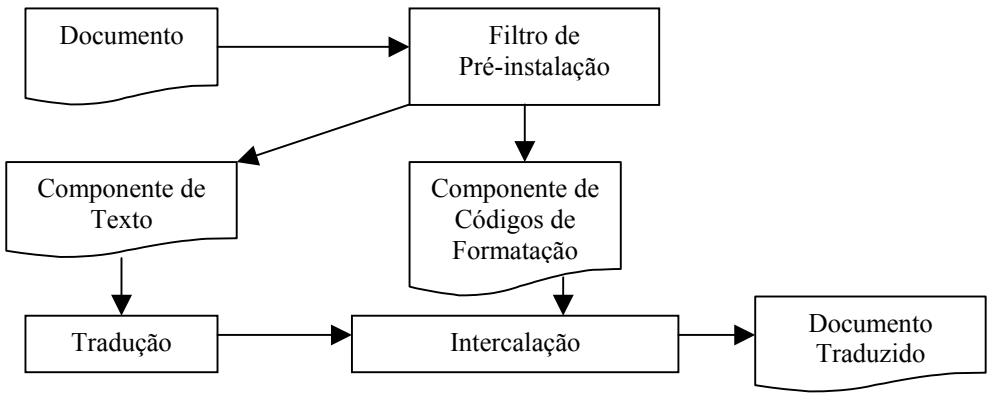


FIGURA 8 - O fluxo de trabalho de uma tradução através de um formato de representação intermediária de dados.

Por exemplo, partindo do pressuposto de que o documento original contém o seguinte fragmento de RTF :

{\plain\cf2\b Hello World }

Poderíamos usar o filtro de pré-tradução para gerar um documento para tradução de seguinte forma:

<1>Hello World<2>

Observe os dois códigos de espaço reservado, <1> e <2>. O filtro de pré-tradução também geraria um documento para armazenar as informações de formatação prontas para serem mescladas após a tradução.

<1> {\plain\cf2\b
<2> }

A tradução então aconteceria, com o cuidado de não mexer nos espaços reservados.

<1> Bonjour le Mond <2>

Finalmente, o filtro de pós-tradução reconstitui o documento, mesclando o conteúdo traduzido com os códigos de formatação.

{\plain\cf2\b Bonjour le Mond }

O uso da XML para notação de documento intermediário para esse tipo de aplicação faz muito sentido, pelas seguintes razões:

- A mãe da XML, a SGML, foi usada no passado para fornecer um denominador comum para a representação de dados em processamento de textos, e criou-se muito conhecimento sobre como lidar com isso, os tipos de elementos necessários etc;
- A XML é uma linguagem amigável “originalmente” com seu suporte nativo para Unicode;
- A capacidade da XML de validar documentos pode ser usada para certificar que os documentos traduzidos não fugiram da estrutura original;
- O escopo significativo para automação no processamento da XML se encaixa bem com a disponibilidade crescente dos sistemas de tradução automáticos e semi-automáticos.

Foi investido muito esforço para assegurar que a XML se tornasse o mais multilíngue possível. O uso do Unicode como codificação de caractere é a prova mais clara desse esforço. Os usuários de XML podem não somente usar todos os recursos do Unicode no conteúdo de seus documentos, como também podem usar na marcação. Em outras palavras, o recurso XML para usar nomes arbitrários para elementos engloba todo o Unicode, caracteres acentuados e outros, e podem ser usados normalmente em nomes de tipos de elementos XML.

As representações intermediárias de dados são uma técnica vital para facilitar o intercâmbio de dados de uma plataforma/pacote de software para outra. Quanto mais poderosa uma determinada ferramenta de computação, maiores chances ela tem de se beneficiar da presença de uma representação intermediária voltada para seu próprio conjunto de recursos em particular. Essa necessidade de funcionalidade específica em uma representação intermediária geralmente leva ao desenvolvimento de ainda uma outra sintaxe para essencialmente os mesmos tipos de informação. RTF, MIF, DCF, CSV, SYLK, DIF, dentre outros, são exemplos de representações de dados de intercâmbio.

Cada nova representação de dados acompanha a necessidade de desenvolvedores de software entendê-la em detalhes e, em seguida, desenvolverem o software de processo para lidar com a nova representação. Os aspectos de baixo nível de processar uma representação, analisando a sintaxe e verificando a estrutura, são típicos e altamente específicos a essa representação. Cada um parece ter seu próprio conjunto de caracteres mágicos, por exemplo. No RTF, caracteres tais como "{" e "}" são especiais. No formato do Framemaker (MIF, Maker Interchange Format), "<" e "\" são especiais e assim por diante. A tarefa em geral é composta de três fases ilustradas a seguir.

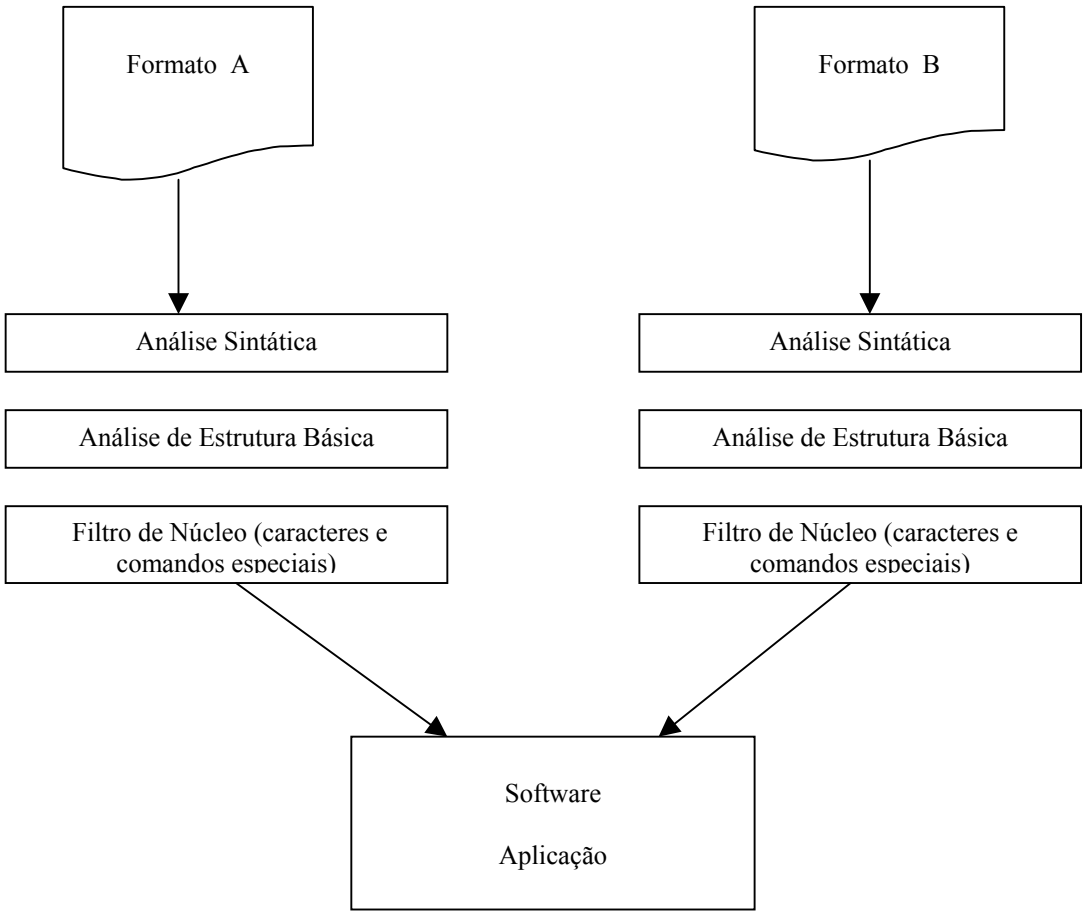


FIGURA 9 - Três estágios de processamento de uma representação intermediária de dados

Desenvolver software para implementar a sintaxe básica e a análise de estrutura dessas representações é um trabalho muito tedioso e complicado.

Precisamos mudar a notação básica para tais representações de intercâmbio um nível acima, de forma que a funcionalidade no que diz respeito à análise de sintaxe, verificação de erro básico etc., possa se tornar um módulo pronto para ser reutilizado várias vezes.

Uma forma de permitir a reutilização de detalhes de baixo nível de análise de uma representação de intercâmbio está em baseá-los todos em uma mesma notação, a XML. Dessa forma, um trecho de código genérico para análise de XML pode ser usado várias vezes, novamente para lidar com detalhes de baixo nível, como é mostrado a seguir.

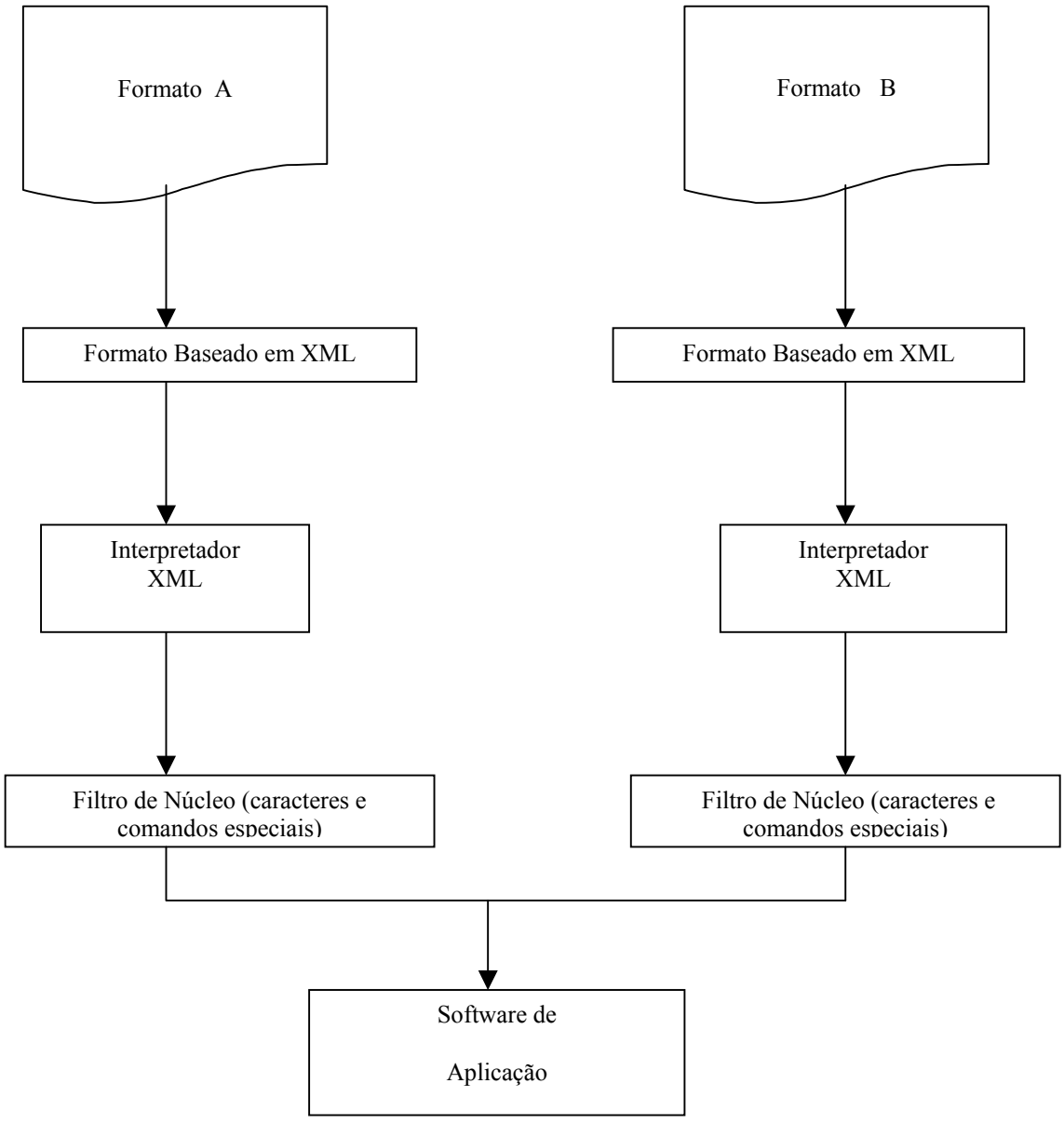


FIGURA 10 - Processamento de representações intermediárias de dados baseados em XML.

O lotus Notes é um bom exemplo de uma poderosa plataforma de computação que se beneficia de uma representação intermediária de dados específica a ele. No Notes, alguns conceitos, como campos de textos compostos, doclinks (vínculo a documentos), campos de valores múltiplos e assim por diante, precisam ser abastecidos de forma compreensiva em uma forma intermediária se for valer a pena usá-los.

Uma aplicação XML, Note Flat File, foi desenvolvida para fornecer tal representação intermediária de dados que possibilite ao Lotus Notes importá-los.

Um seguimento de um arquivo NFF é mostrado a seguir.

```
<NFFBOOK>
  <STYLES>
    <PARAS>
      <PARA AFTER=”1.5” NAME=”Stylel” ALING=”CENTER” LM=”2880”>
    </PARAS>
  </STYLES>
  <NOTE FORM = “Main Form”>
    <ANCHOR NAME = “anchor1”>
    <FIELDS>
    <FIELD NAME = “Age” TYPE = “Number”>
    </FIELDS>
    <COMP NAME = “Body”>
    <P STYLE = “Style1”>
    <TEXT>Hello World</TEXT>
    </P>
    <P STYLE = “Style1”>
    <TEXT>I am a </TEXT>
    <DOCLINK ANCHOR = “anchor1”>
    <TEXT>link.</TEXT>
    </COMP>
  </NOTE>
</NFFBOOK>
```

Um filtro de Importação de Notes padrão foi desenvolvido para entender essa aplicação XML. Ele suporta todos os recursos da plataforma do Notes em termos de campos, formato de texto rico (RTF), doclinks e assim por diante.

Notes Flat File é uma iniciativa da Digitome Eletronic Publishing , em http://www.digitome.com .

Da mesma forma que os autores que usam ferramenta WYSIWYG podem gastar muito tempo formatando o conteúdo que eles criam, os desenvolvedores de software podem gastar muito tempo lutando para implementar rotinas de análise de baixo nível para obter acesso a uma outra representação de intercâmbio definida incorretamente. O uso da XML para essa tarefa significa que a partir de um comando o desenvolvedor do software possui uma vasta gama de experiência e grande quantidade de ferramentas para relacionar-se com a tarefa de processamento de dados. Além disso, como muitos analisadores de XML estão sendo escritos rapidamente com prioridade máxima, cada vez menos boas razões existirão para se desenvolver representações de dados personalizadas. A tendência é usar XML [HAP 99].

Existe uma grande preocupação com a compatibilidade entre as linguagens de programação.

A XML para resolver este problema adotou a Unicode, fazendo assim um intercâmbio entre as diferentes linguagens.

Sendo que outros problemas surgiram, dentre os quais, os chamados caracteres especiais (caracteres que possuem significado especial), estes em cada linguagem de programação possuem funções distintas. Para resolver este tipo de problema a XML faz uma análise sintática, uma análise da estrutura e uma análise destes caracteres (Filtro de Núcleo) para cada documento formado, interpretando e tornando-o compatível com a XML.

Através desta compatibilidade é possível que a XML trabalhe com diferentes tipos de linguagens de programação, sem perder suas características.

3.5 Relação entre características XML, SGML e HTML

- SGML

Vantagens:

- Flexibilidade;
- Adequado para processamento de informações;
- Não-proprietário (não depende de sistemas);
- Reutiliza informações.

Desvantagens:

- Complexidade (o software não é simples de programar);
- Exemplos precisam de DTD, folhas de estilo, catálogo;
- Não é liberada facilmente através da rede;
- Pequeno suporte industrial para o usuário.

- HTML

Vantagens:

- Fácil de usar (proliferação de páginas Web);
- Bom suporte industrial para o usuário;
- Autores escrevem páginas mostrando informações;
- Portabilidade e liberdade através da rede.

Desvantagens:

- Apresenta tags fixas;

- Conteúdo e apresentação misturados juntos;
- Armazenagem de muitas informações pobres;
- Informações armazenadas em HTML e convertidas em SGML.

- XML

Vantagens:

- Extensível;
- Não há grupos fixos de tags;
- Bom para processamento de informações (igual a SGML);
- Adequado para rede;
- Não é tão complexo como a SGML.

Desvantagens:

- Poucos recursos ;
- Poucos Browsers que suportam esta tecnologia

3.6 XML e a Web

Visto que a linguagem XML é bastante direta, não é provável o surgimento de Browsers específicos, já que os atuais precisam apenas de aprimoramento para suportarem ambos, HTML e XML. Browsers que suportam XML:

- Internet Explorer 4.0 (com falhas);
- Internet Explorer 5.0.

Publicador (publisher) é qualquer indivíduo ou organização que entregue conteúdo na Web.

O benefício mais óbvio da XML é que ela possibilita a entrega confiável de qualquer tipo de informação estruturada por toda a Web.

Grupos de colaboração estão propondo aplicação XML padronizada para dados push, metadata e assim por diante.

A XML pode ser usada para dar suporte à descoberta de recursos. Esse é um método de baixo custo para fazer com que informações relevantes sejam encontradas.

Do ponto de vista do cliente, receber XML em vez de HTML torna-o muito mais autosuficiente. Até se os clientes estão recebendo somente informações de âmbito geral, seus processadores XML genéricos lhes permitirão carregá-la, usando uma tabela virtual de conteúdo gerada pelo próprio documento.

Porém, quando a informação codificada em XML que é entregue está relacionada a uma aplicação específica de interesse do cliente, seu valor aumenta dramaticamente..

A XML é capaz de agir como um formato de permuta para documentos e informações de banco de dados. No caso de documentos, fontes com uma estrutura bem definida podem agora ser distribuidos em uma forma que respeita aquela estrutura.

Informação de banco de dados pode ser empacotada em um formato XML para entrega, e desempacotada quando chegar ao seu destino.

A XML pode ser produzida com um simples editor de textos.

Alguns Deles:

- Near & far designer (www.microstar.com) – Utilizado apenas para a autoria de DTDs, ele não serve para criação de documentos XML. Ele permite converter DTDs SGML em DTDs XML.
- Adept-Editor (www.arbotext.com) – Uma ferramenta completa para criação de documentos XML e SGML com a facilidade de apenas apontar e clicar. Pode-se converter documentos SGML para XML e vice-versa.
- FrameMaker + SGML 5.5 (www.brasil.adobe.com) – Permite a editoração de documentos em HTML, PDF, PostScript, SGML e XML .
- XML Pro (www.vervet.com) – Um editor XML de primeira linha, mas não complexo. Com ele, pode-se criar e editar documentos utilizando menus e telas intuitivas e bem organizadas.
- ML NotPad: da Microsoft, versão beta 1.5 –informações no endereço (http://msdn.microsoft.com/xml/notpad/intro.asp)
- XED (www.ltg.ed.ac.uk/~ht/xed.html) – É um simples editor XML escrito em C, plataformas: Win32, Linux, FreeBSD e Solaris.

3.7 Alguns Softwares para XML

Os programas e softwares descritos abaixo, são verdadeiramente aplicações XML.

- Jumbo – É um grupo de classes Java projetado para exibir aplicações XML. O Jumbo foi uma das primeiras aplicações XML. A ferramenta de exibição pode ser usada como uma aplicação independente (standalone) (sob um interpretador Java) ou como um applet sob um exibidor applet Java ou qualquer browser Web com capacidade para Java. Pode ser obtido gratuitamente pelo endereço http://www.venus.co.uk/omf/cml;
- Lark – É um processador XML escrito em Java. Pode ser obtido gratuitamente pelo endereço http://www.textuality.com/Lark;
- LT XML – É um conjunto de utilitários do tipo linha de comando para processamento de documentos XML bem formados;
- MSXML – É um parser XML escrito em Java. O parser carrega os documentos XML e constrói uma estrutura em forma de árvore de objetos de elementos que podem ser manipulados através de um simples grupo de métodos Java. Pode ser obtido gratuitamente pelo endereço http://www.microsoft.com/standarts/xml;
- NXP – É um parser completo e validador XML escrito em Java. Pode ser obtido pelo endereço http://www.edu.uni-klu.ac.at/~nmikula/NPX/;

- TclXML – É uma ferramenta que permite analisar documentos XML e DTDs. Disponível em http://tcltk.anu.edu.au/XML;
- XDK – Desenvolve Kits C++ e Java incluindo parser bem formados e validadores para validar, carregar e acessar documentos XML;
- XMLLINK – É um grupo de vários programas de utilidade Java para a análise e processamento de documentos XML. Disponível em http://www.w3.org/XML/notes.html.

4 Esquemas XML

A necessidade de se criar regras bem formadas para receber e enviar informações de/para uma base de dados originou o surgimento de esquemas.

Os esquemas XML são documentos controladores, controle este, executado por regras estabelecidas pelas normas de segurança e integridade do referido banco de dados. De posse destas regras o programador constrói o esquema XML, que irá interagir junto ao banco de dados XML e o usuário, não permitindo dados indesejáveis e ajudando o mesmo nas tomadas de decisão.

O propósito de um esquema XML é definir e descrever uma classe de documentos XML, desta forma englobando:

- O significado do documento;
- As relações existentes;
- Os tipos de dados;
- Os elementos e conteúdos;
- Os atributos e seus valores;
- As entidades e seus conteúdos;
- As anotações.

Um esquema também tem por objetivo gerenciar informações implícitas das quais os valores não forem informados, definindo um valor para substituição.

Qualquer aplicação que utilize um documento XML bem formado (livre de erros), pode usar um esquema XML para formatar e controlar a entrada e saída dos dados, proporcionando segurança aos mesmos.

Um esquema XML possui uma sintaxe baseada na XML, por definição usa a forma de marcação de um documento XML.

O esquema XML assim como a XML usa muitas das vantagens da DTD (Document Type Definition), na definição inicial da especificação de um esquema para definir um modelo XML.

As DTDs são opcionais e devem ser atreladas ao esquema, elas possuem alguns defeitos:

- Não usam a sintaxe da XML;
- Não dão suporte a tipos de dados;
- Dentre outros.

Sendo assim, o esquema XML , é uma forma poderosa e extensível para descrever as regras para o conteúdo de um documento usando o próprio XML como gramática.

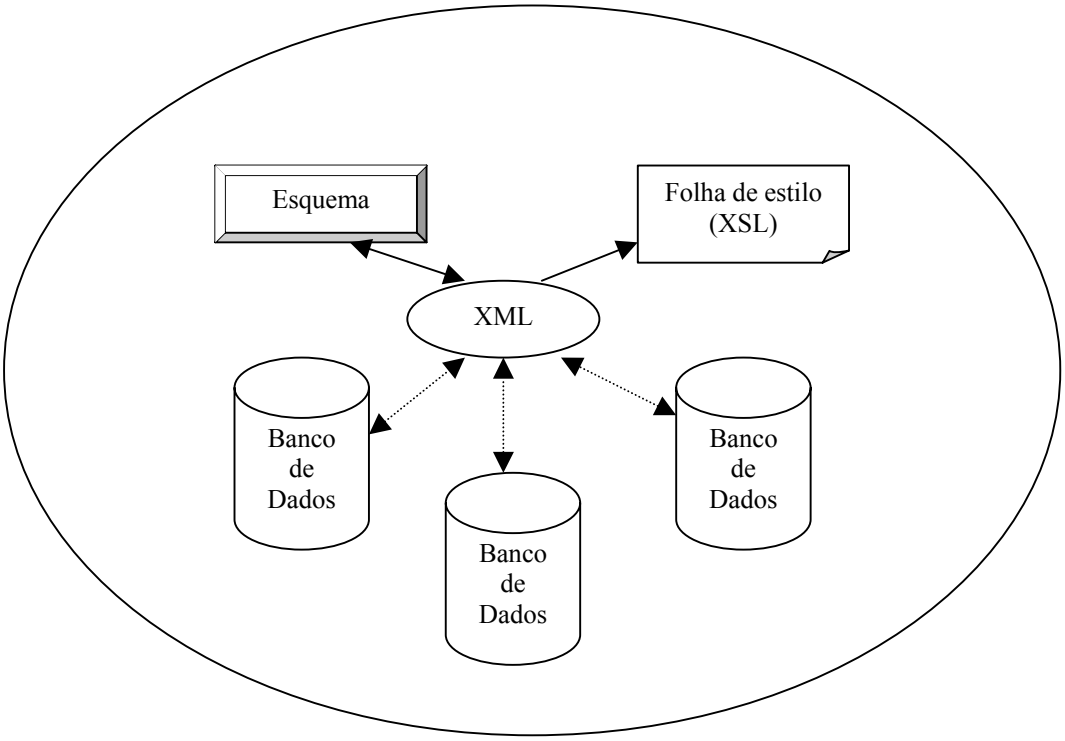


FIGURA 11 - A XML interage com um ou mais bancos de dados, distintos, estes dados são formatados pelo esquema, e posteriormente exibidos pela XSL.

4.1 Tipos de Elementos

Em um esquema temos vários tipos de elementos (Elementtype), dentre os quais, os mencionados a seguir:

- Element

Um elemento pode conter uma referência a um atributo de um outro elemento, sendo que um elemento pode ser requerido ou opcional conforme for indicado pela cláusula "occurs" que é um comando que possui 4 (quatro) estados possíveis :

- Required (requerido);
- Optional (opcional);
- Zeroormore (zero ou mais);
- Oneormore (um ou mais).

Onde o Required é o default.

- EMPTY, ANY e STRING

Os comandos EMPTY de significado “vazio” e o ANY de significado “qualquer conteúdo”, são usados quando pré-definimos os elementos, sendo que o EMPTY pode ser omitido.

A STRING significa qualquer caracter string que não contenha elementos conhecidos como comandos com funções próprias “PCDATA”.

Estes comandos citados podem ser representados de uma forma misturada (juntos) sem que percam suas características.

- GROUP (Grupos)

Indica um conjunto, ou um grupo de elementos ordenados para que possamos designar um atributo aos mesmos , o grupo é tratado como um todo semelhante a um elemento, o comando utilizado para formar este grupo é o GROUPORDER.

- Modelos Abertos e Fechados

O default são os modelos abertos, que permitem que instâncias de elementos e atributos não declarados no elementtype sejam apresentados sem problemas.

Já o modelo fechado só permite que sejam apresentados os elementos ou atributos que estejam declarados no elementtype, tornando-o mais restrito.

- Valores Default

Quando não for especificado um valor este usará o seu valor default, sendo que existem recursos para que possamos definir o valor default para um determinado elemento, conforme a necessidade a qual se destina.

- Aliases e Correlativos

É usado para fazer a equivalência entre dois valores, como por exemplo idiomas diferentes book em inglês é equivalente a livro em português.

<ELEMENTTYPEEQUIVALENT ID=”Livro” TYPE =”#Book”/>
<ELEMENTTYPEEQUIVALENT ID=”Auteur” TYPE =”#Autor”/>

- **Cardinalidades**

Não deve ser usada a cardinalidade de n-1 e sim a de 1-n caso ocorram múltiplas ocorrências de elementos. Por exemplo um pai pode ter vários filhos, sendo esta a relação devemos formar a cardinalidade de modo que seja representada no formato de 1-n.

- **Elementos usados em outros esquemas**

Um esquema pode usar elementos e atributos de outros esquemas. Por exemplo, um sub-elemento nomeado http://pessoas.ore/date pode ser usado dentro de um elemento pessoas.

Desta forma podemos fazer uma relação entre esquemas diferentes que encontram-se em endereços distintos.

- **Atributos como elemento específico da XML**

A XML permite que certas propriedades possam ser expressas em uma forma chamada atributos.

O elementtype pode conter atributos de declaração, os quais são divididos em atributos com valores "enumerated ou notation", dentre outros tipos.

Os atributos notation e enumerated permitem valores aos atributos de uma lista de valores permitidos. Estes são requeridos por ATTTYPE.

Usando o ATTTYPE os atributos requeridos podem ser ENUMERATION ou NOTATION, se não forem proibidos. Nestes casos se o default for especificado, este valor deve seu um dos estipulados.

- **Declaração externa de tipo de elemento (EXTDCLS)**

Embora seja permitido a inclusão de uma entidade externa, recomendamos uma declaração diferente para modularização de esquemas. A declaração de "extdcls" possui um mecanismo limpo para importar (fragmentos) de outro esquema.

Se refere ao texto pela combinação de SYSTEMID e PUBLICID incluído no esquema no lugar do elemento EXTDCLS, e esse texto de substituição está então sujeito às mesmas restrições de validade e interpretação como o resto do esquema.

Em muitos casos o efeito desejado pode ser apresentado melhor através de elementos de referências (e atributos) de outro esquema, ou sub-classe deles.

- Tipos de Dados (Datatype)

O tipo de dado "Datatype" indica que podem ser interpretados os conteúdos como um número, data, etc. O tipo de dado indica que os conteúdos do elemento podem ser analisados gramaticalmente, isto é , distinguimos o tipo de um elemento através de seu Datatype.

Por exemplo 19650813 "13 de agosto de 1965" na ISO 8601 formato de data. Se o conteúdo de um "size" (tamanho) é um inteiro, significa que deveria ser verificado gramaticalmente de acordo com as regras, analisando de forma numérica, pois deverá armazenar em um formato de inteiro.

Em alguns contextos um API pode expor isto como um inteiro em lugar de uma string.

```
<item>
   <names>SHIRT</name>
   <size>8</size>
</item>
```

Há dois contextos principais para Datatypes:

- Quando lidamos com banco de dados APIs, como ODBC, todos os elementos com o mesmo nome contêm o mesmo tipo de conteúdo tipicamente. Por exemplo todos os tamanhos (size) contêm inteiros ou todos os aniversários contêm datas;
- Através do contraste, o tipo do conteúdo pode variar amplamente de instância a instância. Por exemplo, o tamanho (size) pode conter o inteiro 13, ou a palavra aniversário ou até mesmo uma fórmula para calcular o tamanho.

- Tipos de dados complexos

O programa básico XML API expõem todos os conteúdos dos elementos como strings, sem levar em consideração os atributos de tipos de dados. Uma interface ODBC pode usar o atributo tipo de dado (Datatype) para expor cada tipo de elemento em uma coluna, com as colunas por tipo de dado, determinando o tipo de elemento nela contido.

Se um tipo de dado requer uma estrutura complexa para armazenamento, ou um armazenamento baseado no objeto, este será tratado pelo atributo de DT:DT, onde o formato de armazenamento do Datatype pode ser uma estrutura Java Class, COM++ Class, dentre outras. Por exemplo, se uma aplicação precisasse ter um elemento armazenado em uma estrutura "SCHEDULEITEM" e usar um formato privado, ficaria assim.

```
<WHEN DT:DT="ZOO:SCHEDULEITEM">M*D1W4B19650822;100</WHEN>
```

4.2 Exemplo de Esquema XML

Exemplo de um esquema que trabalha com um arquivo em XML, tratando dados em uma relação entre fornecedores e mercadorias, utilizando os recursos citados acima.

XML

```
<?xml:namespace name="http://company.com/schemas/mercadorias/" as="bk"/>
<?xml:namespace name="http://www.ecom.org/schemas/dc/" as="ecom" ?>

<bk:MercFornec>
   <Pessoa>
      <nome>Ricardo Bastos</nome>
      <idade>1965</idade>
   </Pessoa>

   <Pessoa>
      <nome>Francisco Tavares</nome>
   </Pessoa>

   <Pessoa>
      <nome>João Batista</nome>
      <idade>1963</idade>
   </Pessoa>

   <Pessoa>
      <nome>Marta Costa</nome>
   </Pessoa>
   <Mercadoria>
      <fornecedor>Ricardo Bastos</fornecedor>
      <fornecedor>Francisco Tavares</fornecedor>
      <tipo>Parafuso</tipo>
   </Mercadoria>

   <Mercadoria>
      <fornecedor>Marta Costa</fornecedor>
      <tipo>Chapa de Aço</tipo>
      <ecom:preço>110.00</ecom:preço>
   </Mercadoria>

   <Mercadoria>
      <fornecedor>João Batista</fornecedor>
      <tipo>Pregos</tipo>
   </Mercadoria>
</bk:MercFornec>
```

Esquema para http://company.com/schemas/mercadorias:

```
<?xml:namespace name="urn:uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882/" as="s"/?>
<?xml:namespace href="http://www.ecom.org/schemas/ecom/" as="ecom" ?>

<s:schema>

<! – Item nº 1 -- >
   <elementType id="nome">
      <string/>
   </elementType>

<! – Item nº 2 -- >
   <elementType id="idade">
      <string/>
      <dataType dt="date.ISO8601"/>
   </elementType>

<! – Item nº 3 -- >
   <elementType id="Pessoa">
      <element type="#nome" id="p1"/>
      <element type="#idade" occurs="OPTIONAL">
         <min>1950-01-01</min><max>1985-01-01</max>
      </element>
      <key id="k1"><keyPart href="#p1" /></key>
   </elementType>

   <elementType id="fornecedor">
      <string/>
<! – Item nº 4 -- >
      <domain type="#Mercadoria"/>

<! – Item nº 5 -- >
      <foreignKey range="#Pessoa" key="#k1"/>
   </elementType>

   <elementType id="FornecMerc">

<! – Item nº 6 -- >
      <element type="#fornecedor" occurs="ONEORMORE"/>
   </elementType>

<! – Item nº 7 -- >
   <elementType id="Mercadoria" >
      <genus type="#FornecMerc"/>
      <superType href=" http://www.ecom.org/schemas/ecom/commercialItem"/>
      <superType href=" http://www.ecom.org/schemas/ecom/inventoryItem"/>
      <group groupOrder="SEQ" occurs="OPTIONAL">
```

```
        <element type="#fabricanete"/>
        <element type="#fone"/>
      </group>
      <element href="http://www.ecom.org/schemas/ecom/fabric"/>
      <element href="ecom:fabricante"/>
   </elementType>

<! – Item nº 8 -- >
   <elementTypeEquivalent id="Lamina de Aço” type="#Chapa de Aço"/>

</s:schema>
```

Este esquema estabelece uma relação entre o fornecedor e as mercadorias, da seguinte forma:

Tomando como base o arquivo XML onde constam o nome dos fornecedores, o tipo de mercadorias e opcionalmente a idade do fornecedor e o preço da mercadoria, foi montado um esquema para estabelecer regras para estes dados.

O esquema trabalha com estes dados da seguinte forma:

1) Estabelece que o nome será uma string;
2) Que a idade será uma string no formato data conforme estabelecido pela ISO 8601;
3) Que a pessoa possui um nome que é uma string conforme item número 1 e sua idade deve estar compreendida entre 01/01/1950 e 01/01/1985 (sendo este item opcional);
4) O fornecedor somente pode ser usado dentro do elemento Mercadoria ou de um sub-elemento de Mercadoria;
5) O fornecedor é equivalente ao elementtype pessoa no que diz respeito a possuir um nome;
6) FornecMerc é do tipo fornecedor e pode ter um ou mais;
7) Na Mercadoria:
   - O fornecedor herda o conteúdo de outras superclass;
   - Um grupo é formado opcionalmente na sequência mostrada, primeiro o fabricante e depois o fone.
8) A Mercadoria Lâmina de Aço é equivalente a Chapa de Aço.

5 Sistema de Informações da A.R.C.O. e os Padrões Raciais dos Ovinos

A Associação Brasileira de Criadores de Ovinos (Assistência aos Rebanhos de Criadores de Ovinos – A.R.C.O.), fundada em 18 de janeiro de 1942, com a denominação de Associação Riograndense de Criadores de Ovinos, é uma sociedade civil, sem fins lucrativos, de âmbito nacional, designada pela sigla A.R.C.O., nos presentes estatutos, tendo como:

- Sede a cidade de Bagé, Estado do Rio Grande do Sul;
- Área de ação, em todo o Território Nacional;
- Foro Jurídico na Comarca de Bagé, RS;
- Prazo de duração indeterminado.

A A.R.C.O. tem por finalidade:

- Congregar em uma única entidade, todas as associações estaduais de criadores de ovinos e associações promocionais de raças existentes, e que venham a ser fundadas no país, visando manter uma orientação uniforme que propicie a expansão e o desenvolvimento da ovinocultura nacional;
- Lutar pelo aprimoramento das diversas raças aqui criadas, para formar um patrimônio genético próprio, com animais perfeitamente adaptados às condições brasileiras;
- Incentivar a criação de ovinos em todo o território nacional, apoiando às associações existentes e formando novas a fim de poder prestar uma assistência técnica direta a todo ovinocultor;
- Treinar e credenciar técnicos, eng. agrônomos, veterinários e zootecnistas, para executarem os serviços oferecidos pela A.R.C.O., capacitando-os para exercerem essas atividades nas associações estaduais ou promocionais de raça;
- Supervisionar, orientar seus associados e emitir parecer em todos os processos de importação de ovinos e seus materiais de multiplicação;
- Organizar tecnicamente todas as exposições de ovinos em todo o território nacional;
- Firmar convênios com entidades públicas ou privadas, nacionais ou estrangeiras a fim de desenvolver trabalhos em conjunto beneficiando a criação nacional.

A A.R.C.O., executará o SERVIÇO DE REGISTRO GENEALÓGICO DE OVINOS NO BRASIL – S.R.G.O.B., de todas as raças da espécie ovina em todo o território nacional, registrada que é no Ministério da Agricultura e Reforma Agrária – M.A.R.A., como ENTIDADE DE ÂMBITO NACIONAL, nos termos de Portaria nº 47 de 15/10/87 da secretaria nacional de produção agropecuária, e de acordo com o respectivo regulamento do serviço de registro genealógico aprovado pelo ministério da agricultura e reforma agrária.

Para que o criador se torne associado da A.R.C.O. (Associação Brasileira de Criadores de Ovinos) será necessário que o mesmo solicite na referida entidade a ficha cadastro de associados (Anexo 1), que deverá ser preenchida, datada, assinada e após devolvida à A.R.C.O., onde passará em reunião de conselho. Sendo aprovada, a ficha cadastro será enviada ao setor responsável pela digitação para que os dados citados na mesma sejam digitados no sistema, passando dessa forma a fazer parte do banco de dados da entidade.

Na referida ficha deverão constar os dados pessoais do criador, como pessoa física ou jurídica, dados do estabelecimento, se houver, visto que o pecuarista poderá criar em estabelecimento de outro criador ou em seu próprio estabelecimento, podendo inclusive, ter vários estabelecimentos de sua propriedade.

Também deverá ser preenchido na mesma, o afixo, que é o nome exclusivo que identificará os animais do criador, não podendo constar no Registro Genealógico de Ovinos (RGO) o mesmo afixo para outro associado.

Em relação ao afixo, devemos salientar, ainda, que ele deverá ter no máximo duas palavras.

Caso o criador seja uma pessoa menor de 18 anos de idade, será necessário que o mesmo tenha um responsável que deverá preencher os dados pessoais constantes na ficha cadastro, assinando-a posteriormente.

O critério para seleção dos ovinos, baseado na orientação tecnicamente estabelecida, assenta sobre as análises das características de: tipo racial, tipo zootécnico, conformação, constituição, avaliação da lã e detalhes secundários.

Puros de Origem(PO): serão inscritos como PO os produtos nascidos ou não no Brasil, que sejam originários de pais também PO com documentação que comprove a origem. Esta categoria terá como identificação dos animais a tatuagem com o símbolo oficial “A.R.C.O.”.

Raças Exóticas – são consideradas exóticas as raças oriundas de fora do Brasil. Atualmente tem livro de Registro Genealógico as seguintes raças: Merino Australiano, Ideal, Corriedale, Romney Marsh, Texel, Hampshire Down, Ile de France, Suffolk, Lincoln, Border Leicester, Southdown, Lacaune, Wiltshire Horn.

Os animais registrados recebem tatuagens na face interna das orelhas e /ou virilhas, com tintas especiais.

Os algarismos determinam a identificação dos produtos em série numérica progressiva para cada criador, iniciando em 01 ou 001, não podendo haver interrupção, mas podendo utilizar os números pares para fêmeas e ímpares para machos ou vice-versa.

Outra identificação será a do criador, através do código de rebanho, que constará de uma série numérica para cada raça seguida de uma letra em ordem alfabética, até o número 99, repetindo-se o processo com a letra seguinte.

O número que identifica o animal será na orelha esquerda e /ou virilha direita. O código do rebanho na orelha direita.

Desde que seja utilizada a tatuagem de números e que sejam em ordem progressiva, poderá o criador adotar outro sistema de identificação, sendo necessário, neste caso, a solicitação prévia ao conselho deliberativo técnico da A.R.C.O., informando qual o sistema a ser utilizado.

Cada ovino possuirá um criador e um proprietário, uma vez que ao nascer o produto, o criador e o proprietário são a mesma pessoa, sendo que o produto ao ser vendido passa a ter um novo proprietário.

As transferências de ovinos registrados, serão efetivadas mediante autorização assinada e datada pelo criador ou seu representante legal.

Deverá constar, necessariamente, na Autorização de Transferência: Raça, nome do ovino, tatuagem e número de inscrição no S.R.G. (Serviço de Registro Genealógico) dos ovinos transferidos; nome e endereço postal do criador a que é feita a transferência; nome, estabelecimento e município do criador que solicita a transferência .

As autorizações de transferências deverão ser acompanhadas do respectivo certificado de registro ou de controle de genealogia do ovino a ser transferido.

5.1 Padrões Raciais dos Ovinos

A A.R.C.O. controla várias raças com características distintas, tais como:

- Merino Australiano

Aspecto Geral – A rainha das Raças produtoras de lã. É um animal imponente, de aspecto nobre. Bom desenvolvimento corporal. Constituição robusta. Conformação angulosa. Denota grande volume de lã. Raça especializada na produção de lã fina, apresenta um equilíbrio zootécnico orientado 80% para a produção de lã fina e 20% para a carne.

- Ideal

Aspecto Geral – O ideal é uma raça orientada mais no sentido da produção de lã, portanto com mais ênfase para os caracteres laneiros; o seu equilíbrio zootécnico é orientado 70% para a produção de lã e 30% para a carne.

É um ovino de porte médio, bem constituído, denotando vivacidade e vigor, ostentando um velo volumoso.

A sua conformação é bem equilibrada e denota bem suas aptidões de rusticidade e produção de lã fina.

- Corriedale

Aspecto Geral – O ovino Corriedale tem que ter bom porte e deve dar a impressão de um animal de grande vigor e ótima constituição, que se manifesta em sua conformação própria para a produção de carne e lã. Deve ostentar um andar ágil e de grande vitalidade, o que lhe confere uma boa capacidade de deslocamento. Sendo um ovino de duplo propósito, com um equilíbrio zootécnico orientado 50% para a produção de lã, deve ser um animal muito equilibrado, apresentando um esqueleto bem constituído e um velo pesado, extenso e de boa qualidade.

- **Romney Marsh**

Aspecto Geral – O Romney Marsh deve ter o aspecto geral de um animal compacto, vigoroso e bem implantado, denotando vivacidade e nobreza racial. Sendo uma raça desenvolvida e aperfeiçoada mais para a produção de carne, deve ser grande, com boa carcaça, possuindo membros fortes e vigorosos.

É portanto uma raça de duplo propósito, apresentando um equilíbrio zootécnico orientado 60% para a produção de carne e 40% para produção de lã grossa.

A conformação carniceira e a constituição robusta são portanto os principais atributos que o ovino Romney Marsh deve ostentar. Deve ainda apresentar desenvoltura no caminhar.

- **Ile de France**

Aspecto Geral – É um ovino de grande formato, constituição robusta e conformação harmoniosa, típica do animal produtor de carne.

Atualmente é considerada uma raça de duplo propósito, com um equilíbrio zootécnico orientado 60% para a produção de carne e 40% para a produção de lã.

- **Texel**

Aspecto Geral – Ovino de tamanho médio, tendendo para grande, muito compacto, com massas musculares volumosas e arredondadas, constituição robusta, evidenciando vigor, vivacidade e uma aptidão predominantemente carniceira. Atualmente é considerada uma raça de carne e lã, pois a par de uma carcaça de ótima qualidade e peso produz ainda apreciável quantidade de lã.

- **Hampshire Down**

Aspecto Geral – Ovino de tamanho grande, conformação harmoniosa e constituição robusta, compacto e musculoso, evidenciando a primeira vista grande definição racial e sua especialização como produtor de carne. É um animal que denota vivacidade, agilidade e desembaraço.

- **Suffolk**

Aspecto Geral – O Suffolk é um ovino de grande desenvolvimento corporal, de constituição robusta e de conformação tipicamente carniceira. O seu corpo comprido e musculoso, as extremidades desprovidas de lã e revestidas de pelos negros e brilhantes;

a postura de sua cabeça e formato das orelhas, fazem do Suffolk um ovino inconfundível.

Logo a primeira vista o Suffolk impõe a sua condição de raça carniceira.

- Border Leicester

Aspecto Geral – Ovino de grande porte, constituição robusta, muito ágil e levando a cabeça erguida, com um velo de exterior muito característico, deixando totalmente a descoberto a cabeça e os membros dos joelhos e garrões para baixo.

- Karakul

Aspecto Geral – É um ovino de tamanho médio, relativamente pouco pesado, vigoroso, alerta, de conformação muito angulosa, corpo periforme. O trazeiro mais alto e volumoso que o dianteiro, ostentando uma cola muito grossa. Quando adulto tem corpo coberto de pelos compridos e entremeados com lã mais curta, e a cabeça e membros cobertos de pelos curtos e brilhantes.

É uma raça especializada na produção de peles de cordeiros, conhecidas mundialmente pela denominação de “Astrakan”.

- Santa Inês

Aspecto Geral – Ovinos deslanados, de grande porte, mochos, com pelagem variada; machos adultos com 80/100Kg e fêmeas adultas com 60/70Kg.

- Morada Nova

Aspecto Geral – Animais deslanados, mochos, de pelagem vermelha ou branca; machos adultos com 40/60Kg e fêmeas adultas com 30/50Kg.

- Bergamacia Brasileira

Aspecto Geral – Ovinos de grande porte, brancos, mochos, lanudos; machos com 100/120Kg e fêmeas adultas com 70/80Kg.

- Somalis Brasileira

Aspecto Geral – Ovinos de porte médio, deslanados, mochos, cabeça e pescoço negros ou pardos; com anca e cauda gordas; machos adultos com 40/60Kg e fêmeas adultas com 30/50Kg.

- Rabo Largo

Aspecto Geral – Animais de porte médio com cauda de base larga e ponta de lança, deslanados ou com pouca lã, mochos ou chifrudos; machos adultos com 45/50Kg e fêmeas adultas com 30/40Kg.

5.2 Registro de Animais

- Animais PO

Após os nascimentos dos animais PO (puro de origem) o criador preencherá a notificação de nascimento PO (Anexo 2), onde além dos dados pessoais do criador e nome do técnico constarão os dados dos animais, tais como: raça, nome do produto, tatuagem (em seqüência ordenada), sexo, data de nascimento, pai (tatuagem e registro) e mãe (tatuagem e registro), sendo que o registro FBB (Folk Book Brasileiro) do produto e data das inspeções (ao pé e confirmação) serão preenchidos pela A.R.C.O., assim que os mesmos tenham sido obtidos pelo técnico junto ao criador. Referindo-se, ainda, ao preenchimento, a mesma deverá ser datada e assinada pelo criador. Com relação ao pai do produto, nem sempre o mesmo é propriedade do criador, podendo ter sido emprestado de outra cabanha, ser produto de inseminação artificial, ou ainda, transferência de embrião, ficando o criador muitas vezes dependente da A.R.C.O. quanto à informação da procedência do mesmo. Devido esse problema, a maior parte das notificações de nascimentos enviadas pelos criadores chegam à A.R.C.O. incompletas, dificultando os registros dos produtos, por parte da entidade, onde é fornecido um código (FBB) para cada animal.

A ficha de notificação de nascimento deverá ser enviada à A.R.C.O. até noventa dias após os nascimentos.

De posse da notificação de nascimento PO, com todos os dados preenchidos, o setor de competência da entidade digita-os no sistema para que possam fazer parte do banco de dados, ficando pendente a confirmação dos produtos que deverá ser feita por parte de um técnico credenciado pela A.R.C.O..

Finalmente serão enviados ao criador os certificados de registros correspondentes a cada animal registrado e confirmado.

- Animais Base

Com relação a animais base, serão registrados os produtos que preencherem as exigências raciais, sendo que os animais receberão um FBB (Folk Book Brasileiro) sem que haja a necessidade do conhecimento de sua procedência, ou seja, pai e mãe, devido pertencerem a um rebanho de origem desconhecida pela A.R.C.O., ficando a critério do técnico credenciado pela entidade a aprovação ou não dos critérios raciais.

Posteriormente o técnico preenche a ficha de inspeção do rebanho base (Anexo 3), onde deverá constar os dados do criador e também os dados dos animais, tais como: raça, quantidade de machos e fêmeas apresentados, e o número de animais referentes aos apresentados que foram registrados, ou seja os que se encontram dentro dos padrões raciais. Após a totalização os animais que foram registrados serão divididos em dois grupos, machos e fêmeas, sendo informado em cada grupo a tatuagem e o ano de nascimento de cada produto, devendo ainda, a referida ficha ser assinada pelo técnico e pelo criador.

Preenchida a ficha de inspeção, a mesma deverá ser enviada à A.R.C.O. para que seja feita a digitação dos dados nela constantes e para que ocorra o registro dos produtos junto a entidade. Por fim serão enviados ao criador os certificados de registro base correspondentes a cada animal registrado.

- Informações Complementares

Para indicar a morte de um animal PO ou Base, será preenchida a ficha anexo complementar da notificação de nascimento (Anexo 4), onde constarão os dados do criador, data e assinatura do mesmo. Com relação ao animal deverá constar: raça, nome, tatuagem e o registro das ovelhas, cordeiros ou carneiros.

Este procedimento ocorre para que a A.R.C.O. possa dar baixa nos referidos animais, em seus registros, impossibilitando que os mesmos sejam utilizados posteriormente a não ser no caso de um carneiro que possua sêmen congelado em banco de sêmen, comprovando sua origem.

A referida ficha deverá ser enviada à A.R.C.O., no mínimo uma vez ao ano, juntamente com a notificação de nascimento.

6 Banco de Dados e Esquema XML para a ARCO

6.1 Definição do Banco de Dados e do Esquema XML

Devido às dificuldades encontradas pela A.R.C.O. em contatar com seus associados para desenvolver suas atividades foi proposto aplicar a tecnologia XML, para viabilizar um banco de dados via Internet. Com o uso desta tecnologia, minimizam-se os custos da entidade junto ao produtor, criando um novo elo e propiciando o desenvolvimento da ovinocultura.

Conforme foi constatado anteriormente, o criador deve preencher formulários e enviá-los à A.R.C.O., que é uma entidade de nível nacional e encontra-se situada em Bagé, o que dificulta a comunicação e a obtenção de informações. Devido a esta dificuldade de comunicação e ao grande volume de informações que devem ser trocados entre associado e entidade, ocorrem problemas de formulários mal preenchidos ou parcialmente preenchidos, pois o criador não tem acesso ao banco de dados da A.R.C.O.. Isso acarreta devoluções e prejuízo para a entidade, que deve dispor de um funcionário para completar o formulário ou até mesmo contatar o associado para obtenção de maiores informações.

O motivo da falta de informações de dados por parte do associado ocorre muitas vezes devido ao criador utilizar ovinos que não pertencem ao seu plantel (rebanho de ovinos selecionados). Por exemplo, o criador seleciona em seu plantel algumas ovelhas para pôr em cria, mas não possui o carneiro apropriado para as mesmas. Sendo assim, contata outro criador e compra a cobertura. Ao preencher o formulário de nascimento do produto (Anexo 2), o proprietário do plantel (ovelhas selecionadas) não possuindo os dados referentes ao carneiro, envia o formulário deficiente de informações para a A.R.C.O., muitas vezes até impossibilitando que a mesma registre os produtos, tendo que devolver o formulário de notificação de nascimento ao criador para que o mesmo regularize sua situação.

Este tipo de problema cria um descontentamento do criador junto à A.R.C.O., muitas vezes levando o criador a desligar-se da entidade, o que de certa forma não deixa de ser um prejuízo.

Tendo em vista estes problemas, foi proposto à A.R.C.O. uma forma de minimizar estes prejuízos, fazendo uso de uma tecnologia que encontra-se a disposição de todos, a Internet.

Há necessidade da A.R.C.O. dispor as informações para que o pecuarista possa acessá-las sem ter que deslocar-se até a entidade, tendo em vista que estas informações constarão no banco de dados. Para resolver este problema, foi utilizada uma tecnologia conhecida como XML (eXtensible Markup Language), tecnologia esta que permite disponibilizar o conteúdo de um arquivo de dados via Internet (WWW). Os principais arquivos de dados da A.R.C.O. ficarão disponíveis para pesquisas por parte dos pecuaristas, permitindo que os mesmos pesquisem os dados referentes aos animais registrados na A.R.C.O., independente de quem seja o proprietário. De posse destas informações, o proprietário estará apto a preencher seus formulários adequadamente, pois os dados referentes a seu plantel e todos os outros plantéis controlados pela A.R.C.O. estarão disponíveis, bem como a situação em que cada animal se encontra junto à A.R.C.O.. Por exemplo, se o animal já é registrado (confirmado junto à A.R.C.O.), se está vivo ou morto, dentre outras informações referentes ao ovino mencionado.

Contudo, ao disponibilizarmos os arquivos de dados via WWW (World Wide Web) com o auxílio do XML ficam os dados vulneráveis a qualquer modificação, pois o XML não proporciona segurança aos mesmos.

Para solução deste problema, foram utilizados esquemas XML, que são documentos controladores de arquivos de dados em XML, controle este executado por regras estabelecidas pelas normas de segurança e integridade do referido banco de dados. Possui a característica de ser um arquivo a parte e de utilizar um sintaxe XML.

O esquema XML é formado por regras, regras estas que são as mesmas que controlam o banco de dados primitivo na entidade (A.R.C.O.). De posse destas regras, o programador (desenvolvedor de software) constrói o esquema XML, que irá interagir junto ao banco de dados XML e ao usuário, não permitindo dados indesejáveis e ajudando o mesmo nas tomadas de decisões.

O objetivo de utilizarmos esquemas para fazer a proteção dos arquivos de dados em XML é não somente disponibilizar os dados na WWW, e sim estabelecer um meio de comunicação entre a entidade (A.R.C.O.) e o criador, permitindo ao mesmo além de consultar os dados preencher os formulários (XML) para posterior envio à A.R.C.O..

Com o uso do esquema XML que possui as mesmas regras que controlam o banco de dados primitivo, estará o banco de dados sendo protegido pelo esquema XML que somente permitirá a inclusão de dados realmente compatíveis com os existentes na entidade, não aceitando informações indesejáveis, ou seja, que estejam fora das regras estabelecidas.

A troca de informações entre a A.R.C.O. e o associado se dará da seguinte forma:

Criou-se arquivos XML com a mesma estrutura dos arquivos existentes na A.R.C.O., como segue:

Arquivos da A.R.C.O.		Arquivos equivalentes em XML	
ANIMA_PR.dbf	(Anexo 11) =>	OvinosPO.xml	(Anexo 5)
ANIMA_BA.dbf	(Anexo 12) =>	OvinosBA.xml	(Anexo 6)
FICHARIO.dbf	(Anexo 13) =>	Fichario.xml	(Anexo 7)

Estes arquivos XML serão controlados por regras, as mesmas regras que regem o referido arquivo de dados na entidade (A.R.C.O.), estando estas dispostas em esquemas XML, como segue:

OvinosPO.xml	=>	OvinosPO.biz	(Anexo 8)
OvinosBA.xml	=>	OvinosBA.biz	(Anexo 9)
Fichario.xml	=>	Fichario.biz	(Anexo 10)

Estes arquivos XML serão alimentados com os dados dos arquivos existentes na A.R.C.O. e serão dispostos na WWW, sendo os mesmos protegidos pelas regras dos esquemas XML equivalentes.

Estando estes arquivos disponibilizados na WWW o associado poderá acessá-los para consulta e inclusão de dados.

Para que o associado possa interagir junto a este banco de dados, o mesmo necessitará de um software, que será desenvolvido por um programador de sua

confiança, criando software para consulta, alteração, inclusão e exclusão de dados do arquivo XML. A A.R.C.O., independente do software que atuará junto ao arquivo XML, manterá a segurança e a integridade das informações, pois somente serão aceitas informações que se adequarem as regras estabelecidas pelo esquema XML.

Seguindo uma periodicidade estabelecida pela A.R.C.O., os arquivos XML serão capturados pela entidade (A.R.C.O.) onde serão analisados os novos registros (incluidos pelos criadores) e caso aceitos passarão a fazer parte do arquivo primitivo da entidade, posteriormente atualizando o arquivo XML e novamente o disponibilizando na WWW.

Esta comunicação se dará via WWW, não importando que tipo de hardware, software ou até mesmo plataforma a qual o associado esteja submetido, ficando por conta do associado contratar um programador para que de posse dos recursos disponíveis pelo mesmo (hardware e plataforma) crie um software para interagir seu banco de dados com os arquivos XML disponibilizados pela A.R.C.O..

6.2 Relação entre o Banco de Dados XML e o Esquema XML

O arquivo OvinosPO.xml (anexo 5) foi criado tomando como base o arquivo Anima_PR.dbf (anexo 11) existente na A.R.C.O. . O arquivo criado possui as mesmas características do arquivo original (Anima_PR.dbf) diferenciando-se no nome das tag's que referenciam-se aos dados, bem como pela eliminação de alguns campos que existem no arquivo original que somente são de valia para a A.R.C.O. .

O arquivo criado (OvinosPO.xml) está protegido por um esquema XML denominado OvinosPO.biz (anexo 8), onde constam as regras para manipulação dos dados no arquivo OvinosPO.xml, estas regras estão de acordo com as submetidas ao arquivo original (Anima_PR.dbf).

A relação existente entre o arquivo OvinosPO.xml e o esquema XML OvinosPO.biz se dá da seguinte maneira:

O arquivo OvinosPO.xml (anexo 5) possui na sua primeira linha uma indicação da versão a qual está submetido, na segunda linha ocorre uma indicação de que este arquivo está sendo protegido por um esquema XML e indica o nome deste esquema XML ao qual está submetido. A partir daí surgem as tag's, onde cada uma corresponde a um campo no registro do arquivo original (Anima_PR.dbf), um grupo de tag's distintas formam um registro, desta forma as tag's se repetem para cada registro do arquivo original constituindo um arquivo XML equivalente ao existente na A.R.C.O..

O esquema XML OvinosPO.biz protege o arquivo OvinosPO.xml da seguinte maneira:

O esquema XML OvinosPO.biz (anexo 8) possui na sua primeira linha uma indicação da versão a qual está submetido, a segunda linha é formada por um comentário ilustrativo, a terceira linha indica o nome do esquema XML e as duas linhas subsequentes indicam a relação existente com o browser ao qual está submetido, formando o cabeçalho do esquema, as próximas linhas contém as regras que fazem a proteção do arquivo OvinosPO.xml.

Em uma primeira etapa são descritos os elementos que fazem parte do esquema, bem como o número de vezes que poderão aparecer, desta forma permitindo que se possa definir se uma determinada tag é obrigatória ou opcional, conforme as regras que regem o arquivo original, por exemplo:

- <element Type = “Tatuagem”/>

Deve existir a tag, mas pode não conter informação.

- <element Type = “GeracoesControladas” minOccurs=”0” maxOccurs=”1”/>

Pode existir ou não a tag (opcional).

- <element Type = “CodigoEstabelecimento” minOccurs=”0” maxOccurs=”*”/>

A tag pode não existir ou existir uma ou várias vezes sob o mesmo nome.

Após esta definição, em uma segunda etapa, cada tag é tratada individualmente estabelecendo as regras de formação do conteúdo das mesmas, segundo as regras estabelecidas pela A.R.C.O. , como segue:

- O elemento “FBB” é uma string que somente aceita texto, possui um atributo “codigo” que é uma chave primária e deve obrigatoriamente existir;
- O elemento “Tatuagem” é uma string que aceita somente texto;
- O elemento “DuplaTatuagem” é uma string que somente aceita texto, possui um atributo “Tat” que somente aceita os valores S, N, s, n, tendo por default o “N”;
- O elemento “GeraçoesControladas” é uma string que aceita somente texto;
- O elemento “Confirmado” é uma string que somente aceita texto, possui um atributo “Conf” que somente aceita os valores C, N, R, A, I, c, n, r, a, i, tendo por default o “N”;
- O elemento “Nome” é uma string que aceita somente texto;
- O elemento “Nacionalidade” é uma string que aceita somente texto;
- O elemento “FBBPai” é uma string que aceita somente texto;
- O elemento “RegistroPaiExterior” é uma string que aceita somente texto;
- O elemento “NomePai” é uma string que aceita somente texto;
- O elemento “FBBMae” é uma string que aceita somente texto;
- O elemento “RegistroMaeExterior” é uma string que aceita somente texto;
- O elemento “NomeMae” é uma string que aceita somente texto;
- O elemento “DataNascimento” aceita somente data;
- O elemento “Sexo” é uma string que somente aceita texto, possui um atributo “Sex” que somente aceita os valores M, F, m, f;
- O elemento “CodigoCriador” aceita somente números inteiros;
- O elemento “NomeCriador” é uma string que aceita somente texto;
- O elemento “CodigoAfixo” aceita somente números inteiros;
- O elemento “CodigoEstabelecimento” aceita somente números inteiros;
- O elemento “NomeEstabelecimento” é uma string que aceita somente texto;
- O elemento “CodigoProprietario” aceita somente números inteiros;
- O elemento “NomeProprietario” é uma string que aceita somente texto;
- O elemento “NumeroDocumentConfirm” é uma string que aceita somente texto;
- O elemento “NumeroDocumentNotific” é uma string que aceita somente texto;
- O elemento “DataRegistro” aceita somente data;
- O elemento “VivoMorto” é uma string que somente aceita texto, possui um atributo “Vivo” que somente aceita os valores S, N, s, n, tendo por default o “S”;
- O elemento “DataMorte” aceita somente data;
- O elemento “Transferido” é uma string que somente aceita texto, possui um atributo “Trans” que somente aceita os valores S, N, s, n, tendo por default o “N”;

- O elemento “DataConfirmacao” aceita somente data;
- O elemento “DataAnotacaoConfirmacao” aceita somente data;
- O elemento “CodigoTecnico” aceita somente números inteiros;
- O elemento “NomeTecnico” é uma string que aceita somente texto;
- O elemento “DataUltimaTransferencia” aceita somente data;
- O elemento “DataAnotacaoUltimaTransf” aceita somente data;
- O elemento “TipoCobertura” é uma string que somente aceita texto, possui um atributo “Tipocob” que somente aceita os valores MN, IA, TE, mn, ia, te, tendo por default o “MN”;
- O elemento “TipoParto” é uma string que somente aceita texto, possui um atributo “Tipopart” que somente aceita os valores 1, 2, 3, 4, 5, tendo por default o “1”;
- O elemento “AtaConfirmacao” é uma string que aceita somente texto;
- O elemento “DataPrimeiroCertificado” aceita somente data;
- O elemento “DescricaoRaca” é uma string que aceita somente texto;
- O elemento “DataDigitacao” aceita somente data.

6.3 Implementação

Foi proposto em março de 2000, através do PEP, a implementação de um banco de dados XML compatível com o existente na A.R.C.O. , sendo o mesmo protegido por um esquema XML contendo as mesmas regras que regem o banco de dados primitivo da A.R.C.O., estes dados seriam visualizados via Internet, com auxílio de uma folha de estilo conhecida como XSL. No decorrer do desenvolvimento deste trabalho foi constatado que a visualização seria apresentada com o auxílio do HTML e PHP, em substituição ao XSL, visto que o objetivo principal seria o banco de dados XML e o esquema XML, pois estes seriam tratados por softwares desenvolvidos posteriormente pelo programador da A.R.C.O., desprezando desta forma a metodologia de visualização utilizada.

O processo de visualização se dará de duas formas:

* Primeiro - Por um gerenciador em HTML que tem por objetivo apresentar uma página inicial onde constam a apresentação da entidade A.R.C.O. e link`s para outras três páginas HTML, tendo cada uma por objetivo interagir com arquivos XML distintos.

A comunicação entre uma página HTML e um arquivo XML se dá através de um applet java, denominado XMLDSO.class, que deve constar na página HTML.

```
<applet  code=com.ms.xml.dso.XMLDSO.class
width=0%
height=0
id=xmldso
mayscript=TRUE
>
```

Posteriormente é indicado na página HTML o nome do arquivo XML com o qual irá interagir. (dentro da tag applet)

```
<PARAM  NAME=”url”  VALUE=”Fichario.xml”>
```

Após estabelecida a comunicação entre a página HTML e o arquivo XML, é criada uma tabela para apresentação dos dados.

A tabela HTML é vinculada ao arquivo XML através do nome da tag (do arquivo XML), desta forma tratando-se isoladamente cada tag para apresentação na página.

```
<tr>
  <td><b>Socio</b></td>
  <td><FONT COLOR=”#000000”><div datasrc=#xmldso datafld=Socio></td>
</tr>

<tr>
  <td><b>Nome</b></td>
  <td><FONT COLOR=”#000000”><div datasrc=#xmldso datafld=Nome></td>
</tr>
```

Finalmente foram criados dois botões com a finalidade de navegar pela tabela.

```
<input type=button  value=”Avançar”
        onclick=’xmldso.recordset.moveprevious ();’>
<input type=button  value=”Retornar”
        onclick=’xmldso.recordset.movenext ();’>
```

Desta forma foi possível apresentar a visualização de um arquivo XML com o auxílio de uma página HTML, mostrando os dados em uma tabela com a possibilidade de navegação registro a registro.

***** Segundo - Também com o auxílio de um gerenciador HTML (Arco.html) que tem por objetivo apresentar uma página inicial, com a apresentação da entidade A.R.C.O., possuindo link’s para outras páginas e script’s em PHP que desenvolverão pesquisas em arquivos XML, bem como inclusão de dados em um arquivo XML de cadastro de sócio, que será criado automaticamente, cujo nome deste será o CPF do futuro sócio (“CPF”.xml), para posteriormente ser enviado à A.R.C.O., sendo este último arquivo protegido por um esquema XML contendo as mesmas regras de proteção utilizadas pela A.R.C.O. .

O script GravaFicha.php tem por objetivo criar o arquivo Ficha.xml, onde constarão os dados cadastrais do novo sócio, a ser enviado posteriormente a entidade.

```
<?php

echo "</b> </font>";
```

```
echo '<font color="black" size="2">';
echo '<p align="left">';
$f = fopen("$cpf”.xml","w");
        if (!$f) {
                echo "ERRO AO ABRIR ARQUIVO,<br>\n";
        exit;
        }else{
                fputs($f,"<?xml version=\"1.0\"?>\n");

                fputs($f,"<ARCO xmlns=\"x-schema:Fichario.biz\">\n");
                fputs($f,"<Fichario>\n");

                fputs($f,"<Socio Codigo=\"S\">S</Socio>\n");

                fputs($f,"<Nome>$nome</Nome>\n");
                fputs($f,"<Endereco>$endereco</Endereco>\n");
                fputs($f,"<Bairro>$bairro</Bairro>\n");
                fputs($f,"<CXP>$cxp</CXP>\n");
                fputs($f,"<CEP>$cep</CEP>\n");
                fputs($f,"<Localidade>$localidade</Localidade>\n");
                fputs($f,"<Cidade>$cidade</Cidade>\n");
                fputs($f,"<Estado>$estado</Estado>\n");
                fputs($f,"<DDD>$ddd</DDD>\n");
                fputs($f,"<Fone>$fone</Fone>\n");
                fputs($f,"<Ramal>$ramal</Ramal>\n");
                fputs($f,"<Fax>$fax</Fax>\n");
                fputs($f,"<CPF>$cpf</CPF>\n");
                fputs($f,"<DataNasc>$DtNasc</DataNasc>\n");
                fputs($f,"<Profissao>$profissao</Profissao>\n");

fputs($f,"<Naturalidade>$naturalidade</Naturalidade>\n");
                fputs($f,"<Sexo Sex=\"$sexo\"></Sexo>\n");

fputs($f,"<Representante>$representante</Representante>\n");
                fputs($f,"<DataMatric>$DtMatricula</DataMatric>\n");
                fputs($f,"<Responsavel>$responsavel</Responsavel>\n");

                fputs($f,"<TipoSocio Tiposoc=\"C\"></TipoSocio>\n");

                fputs($f,"<TipoPessoa
Tipopes=\"$tipopessoa\"></TipoPessoa>\n");

                fputs($f,"<CodigoTecnico></CodigoTecnico>\n");

                fputs($f,"<Email>$email</Email>\n");


                fputs($f,"</Fichario>\n");
                fputs($f,"</ARCO>\n");



                fclose($f);
              }

?>
```

As pesquisas aos arquivos de dados XML se darão por intermédio de script’s PHP gerenciados por páginas HTML, abrangendo os arquivos de sócios, ovinos puros de origem (PO) e ovinos base.

Estas pesquisas poderão ser feitas conforme solicitação do usuário, podendo pesquisar através de palavras chaves, o que proporciona grandes vantagens. Os campos de pesquisa disponibilizados são os mesmos cujos arquivos primitivos da A.R.C.O. utiliza, devido sua grande importância.

Tomando como exemplo, a etapa de pesquisa ao arquivo de sócios (Fichario.xml) com auxílio de uma página HTML (MainSO.html) pode-se optar por pesquisar por cidade ou nome do sócio, bastando informar posteriormente a que cidade ou nome refere-se, podendo também optar pela lista completa de todos os sócios.

```
<HTML>
    <HEAD>
    <font color="#ff0000">
         <TITLE>Informacoes sobre os Socios</TITLE>
    </HEAD>
    <BODY BGCOLOR="#d9d9f3" TEXT="#000000" LINK="#0000ff">
         <BR>
         <BR>
<BR><H1><B>Informacoes sobre os Socios</B></H1>
         <BR>
         <BR>
         <BR><HR>
          <H2><B>Pesquisa Socios</B></H2>

     <FORM ACTION="pesquisaSO.php" METHOD=GET>
            <CENTER>
                 <TABLE>
                     <TR>
                          <TD>
                               <B>Pesquisa Socios por</B>
                          </TD>
                          <TD>
                               <SELECT NAME=searchBy  SIZE=1>
                                       <OPTION>Cidade
                                       <OPTION>Nome
                               </SELECT>
                           </TD>
                      </TR>
                      <TR>
                          <TD>
                               <B>Palavra Chave</B>
                          </TD>
                          <TD>
                               <INPUT TYPE=TEXT NAME="searchKeyword">
                          </TD>
                       </TR>
                       <TR>
                           <TD> </TD>
                           <TD>
                                <INPUT TYPE="submit" VALUE="Submit">
                           </TD>
                        </TR>
                  </TABLE>
              </CENTER>
         </FORM>
         <BR><HR>
         <BR>
         <BR>
         <BR>
```

```
         <BR>
         <FORM ACTION="totalSO.php" METHOD=GET>
               <CENTER>
                    <INPUT TYPE="submit" VALUE="Lista completa de
Socios">
               </CENTER>
          </FORM>
    </BODY>
</HTML>
```

A pesquisa se dará através de um script PHP (PesquisaSO.php), caso seja utilizado algum argumento de pesquisa. Este script faz uso de um segundo script PHP (GeralSO.php) que tem por objetivo interagir e obter as informações do arquivo XML.

```
<HTML>
      <HEAD>
            <TITLE>Resultados da Pesquisa</TITLE>
      </HEAD>
      <BODY BGCOLOR="#d9d9f3" TEXT="#000000" LINK="#0000ff">
               <B>Resultados da Pesquisa</B>
               <HR><BR>

       <TABLE BORDER=1>
                 <THEAD>
                      <TR>
                           <TH> Codigo </TH>
                           <TH> Nome </TH>
                           <TH> Endereco </TH>
                           <TH> Bairro </TH>
                                  .
                                  .
                                  .
                           <TH> Email </TH>
                       </TR>
                 </THEAD>
                 <TBODY font bgcolor="#a68064">
                        <?php

require("c:\apache\htdocs\php\ArcoPHP\geralSO.php");
$books = readBookInfo();
if (strcmp($searchBy, "Nome") == 0)   {
   if (($book = searchBookByISBN($books, $searchKeyword)))
                                     printBookInfo($book["Socio"],
                                  $book["Codigo"], $book["Nome"],
                               $book["Endereco"], $book["Bairro"],
                                                  .
                                                  .
                                                  .
                                $book["Tipopes"], $book["Email"]);

                                }

elseif (strcmp ($searchBy, "Cidade") == 0)   {
     for  ($i=0; $i<count($books); $i++)   {
       if (strstr(strtolower(trim($books[$i]["Cidade"])),
                                strtolower(trim($searchKeyword))))
```

```
                                printBookInfo($books[$i]["Socio"],
                                            $books[$i]["Codigo"],
                                                    .
                                                    .
                                                    .
                                            $books[$i]["Email"]);

                                        }
                                    }

                                ?>
                        </TBODY>
        </TABLE>
        </BODY>
</HTML>
```

Se a opção for pesquisar a lista completa dos sócios, um outro script PHP será utilizado (TotalSO.php), com objetivos óbvios, este também fará uso de um segundo script PHP (GeralSO.php) para estabelecer a comunicação com o arquivo XML.

```
<HTML>
            <HEAD>
                        <TITLE>Lista Completa de Socios</TITLE>
            </HEAD>
            <BODY BGCOLOR="#d9d9f3" TEXT="#000000" LINK="#0000FF">
                        <H1>Lista Completa de Socios</H1>
                        <HR><BR>
                        <TABLE  BORDER=1>
                                <THEAD>
                                        <TR>
                                                <TH> Codigo </TH>
                                                <TH> Nome </TH>
                                                        .
                                                        .
                                                        .
                                                <TH> Eamil </TH>
                                        </TR>
                                </THEAD>
                                <TBODY font bgcolor="#a68064">
<?php

require("c:\apache\htdocs\php\ArcoPHP\geralSO.php");
$books = readBookInfo();
        for ($i=0; $i<count($books); $i++)  {
            printBookInfo($books[$i]["Socio"], $books[$i]["Codigo"],
                        $books[$i]["Nome"], $books[$i]["Endereco"],
                                                .
                                                .
                                                .
                        $books[$i]["Tipopes"],$books[$i]["Email"]) ;
            }
?>
                          </TBODY>
```

O script mencionado anteriormente, GeralSO.php, encontra-se separado por servir a dois script's distintos, evitando redundância e aumentando a eficiência.

```
<?php
    $file = "Fichario.xml";
    $currentTag = "";
    $SocioValue = "";
    $CodigoValue = "";
          .
          .
          .
    $books = array();


 function startElement ($parser, $name, $attr) {
       global $currentTag, $CodigoValue, $SexattrValue,
              $TipoSocattrValue, $TipoPesattrValue;
              $currentTag = $name;
       if (strcmp ($name, "Socio") == 0) {
                   $CodigoValue = $attr["Codigo"];
           } elseif (strcmp ($name, "Sexo") == 0) {
              $SexattrValue = $attr["Sex"];
           } elseif (strcmp ($name, "TipoSocio") == 0) {
              $TipoSocattrValue = $attr["Tiposoc"];
           } elseif (strcmp ($name, "TipoPessoa") == 0) {
              $TipoPesattrValue = $attr["Tipopes"];
           }
         }

 function endElement ($parser, $name)  {
          global $SocioValue,$CodigoValue, $NomeValue,
                                  . . .
          $EmailValue, $books, $authorCount, $descriptionValue;
     if (strcmp ($name, "Fichario") == 0)  {
                            $books[] = array ("Socio"=> $SocioValue,
                                                        . . .
                                                "Email" => $EmailValue) ;
                $SocioValue = "";
                $CodigoValue = "";
                    . . .
                $EmailValue = "";

          }
       }


  function characterData($parser, $data)  {
                          global $SocioValue,$CodigoValue, $NomeValue,
                                                    . . .
                                  $EmailValue, $currentTag, $authorCount;
            if (strcmp ($currentTag, "Socio") == 0)  {
               $SocioValue .= $data;
            } elseif (strcmp ($currentTag, "Nome") == 0)  {
               $NomeValue .= $data;
                       . . .
            } elseif (strcmp ($currentTag, "Email") == 0)  {
               $EmailValue .= $data;
            }
       }
```

```
 function readBookInfo()  {
        global $file, $books;
        $xml_parser = xml_parser_create();
        xml_set_element_handler($xml_parser, "startElement",
                                              "endElement");
        xml_set_character_data_handler($xml_parser,
                                       "characterData");
        xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING,
                                                        false);

        if (!($fp = fopen($file, "r")))  {
                 die("Could not open $file for reading");
              }
              while (($data = fread($fp, 4096)))  {
                 if (!xml_parse($xml_parser, $data, feof($fp))) {
                    die(sprintf("XML error at line %d column %d",

                     xml_get_current_line_number($xml_parser),
                     xml_get_current_column_number($xml_parser)));
                  }
              }

              xml_parser_free($xml_parser);
              return $books;
            }


  function printBookInfo($SocioValue, $CodigoValue, $NomeValue,
                                        . . .
                               $TipoPesattrValue, $EmailValue)  {

            print "<TR>";
            print "<TD>$CodigoValue</A></TD>";
            print "<TD>";
            print " $NomeValue ";
             . . .
            print "<TD>$EmailValue</TD>";
            print "</TR>";
       }


  function searchBookByISBN($books, $Nome)  {
       for ($i = 0; $i < count($books); $i++)  {
           if (strcmp(trim($books[$i]["Nome"]), trim($Nome)) == 0)  {
                    return $books[$i];
           }
       }
       return null;
    }
?>
```

Este processo se repete para os arquivos de ovinos puros de origem (PO) e para os ovinos base, diferenciando-se por tratarem-se de arquivos XML distintos, bem como linhas de pesquisas diferenciadas.

6.4 Aspectos de Segurança

A implementação do banco de dados XML inicialmente será a caráter experimental, devido a preocupação da A.R.C.O. com a segurança de seus dados.

Em reuniões feitas junto a representantes da A.R.C.O. e seu programador, concluiu-se que seria apresentado um banco de dados XML que possuísse as mesmas regras de proteção oferecidas pela entidade, regras estas oferecidas pelo esquema XML.

Este banco de dados protegido por regras foi apresentado e aprovado junto aos representantes da A.R.C.O. e seu programador, que pretendem estender esta tecnologia a outros bancos de dados existentes na entidade.

Inicialmente a A.R.C.O. pretende através de seu programador, criar um software para alimentar este banco de dados XML, visto que a entidade continuará a trabalhar com seu sistema primitivo, inicialmente, fazendo uso da tecnologia XML para disponibilizar os dados via Internet.

Dando este passo inicial, a A.R.C.O. pretende desenvolver um software que alimente diversos arquivos XML (protegidos por regras), arquivos estes que atualmente são fichas enviadas ao criador para serem preenchidas manualmente, e através deste software o criador preencherá a ficha e remeterá à A.R.C.O., onde haverá um outro software que fará a validação dos dados para posterior inclusão no banco de dados primitivos.

O software desenvolvido pela A.R.C.O. será comercializado junto ao criador, o que trará benefícios para a mesma, bem como para o criador, pois não mais precisará deslocar-se até a entidade para preencher os formulários, bastando consultar os arquivos disponibilizados pela A.R.C.O., em XML, para obter as informações necessárias e após preencher a ficha correspondente, oferecida pelo software, que por sua vez alimentará um arquivo XML correspondente, este será enviado via Internet para A.R.C.O. onde será validado e incorporado ao banco de dados primitivo.

O principal patrimônio da A.R.C.O. é seu banco de dados, sendo assim ela pretende implantar esta tecnologia aos poucos, mantendo a maior segurança possível, pois hoje em dia a Internet não oferece muita segurança.

A tecnologia foi apresentada e aprovada pela A.R.C.O., dando segurança a validação dos dados, mas não protegendo contra ataques indesejáveis, por este motivo o programador da A.R.C.O. pretende criar softwares validadores provando a autenticidade dos dados, visto que, os mesmos, estão convencidos que o uso desta tecnologia é inevitável para a sobrevivência da entidade.

7 Conclusão

Este trabalho foi desenvolvido junto à A.R.C.O. (Associação Brasileira de Criadores de Ovinos) com objetivo de implantar uma tecnologia conhecida como XML, que trará grandes avanços tecnológicos para esta entidade, bem como servir de apoio para o desenvolvimento do setor primário, visto ser este a principal atividade econômica de nossa região.

Três arquivos foram utilizados na aplicação desta tecnologia, arquivos estes de significativa importância no banco de dados da entidade.

O principal objetivo deste trabalho é disponibilizar estes arquivos na Internet utilizando uma estrutura equivalente à existente, sendo estes arquivos (XML) protegidos por um esquema XML que contém as mesmas regras de formação que regem os arquivos do banco de dados da A.R.C.O. .

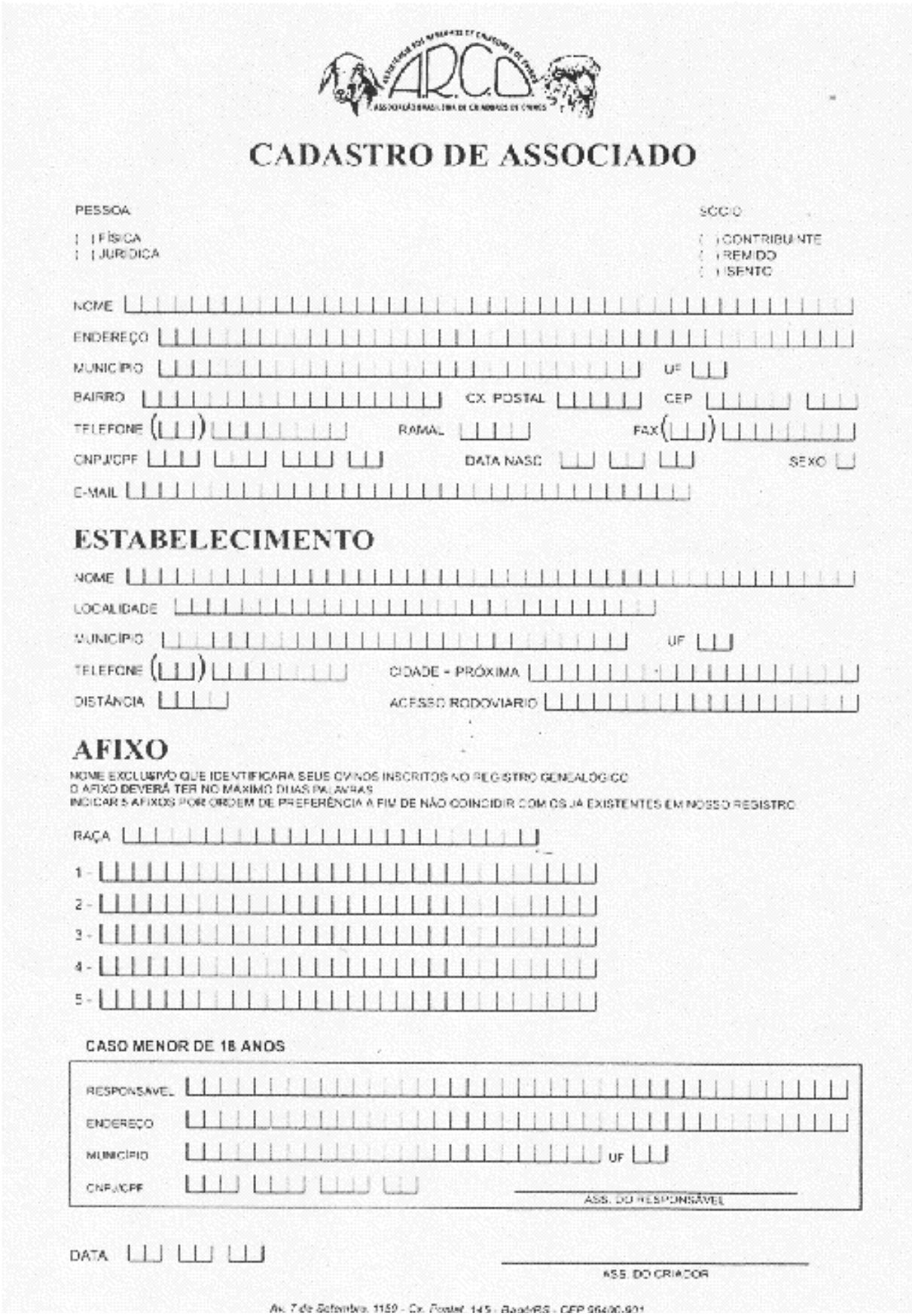
O uso desta tecnologia permitirá aos associados uma maior interação junto à A.R.C.O., ficando por conta dos mesmos a obtenção de softwares que proporcionem o manuseio destes arquivos via Internet, independente de linguagem de programação e/ou plataforma à qual estejam submetidos.

A A.R.C.O. observará vantagens tais como:

- Disponibilizar os dados a um baixo custo;
- Fortalecer o vínculo (comunicação) junto ao associado;
- Obter informações do associado, antes possíveis somente através de formulários impressos;
- Maior desenvolvimento da ovinocultura;
- Possibilidade de aplicar esta tecnologia a outros arquivos;
- Atualização periódica do banco de dados;
- Criar um vínculo com o associado para aplicação de outras tecnologias baseadas na Internet.

Anexo 1

Cadastro de Sócios



A.R.C.O.

ASSOCIAÇÃO BRASILEIRA DE CRIADORES DE OVINOS

CADASTRO DE ASSOCIADO

PESSOA

( ) FÍSICA
( ) JURÍDICA

SÓCIO

( ) CONTRIBUINTE
( ) REMIDO
( ) ISENTO

NOME

ENDEREÇO

MUNICÍPIO UF

BAIRRO CX. POSTAL CEP

TELEFONE ( ) RAMAL FAX ( )

CNPJ/CPF DATA NASC SEXO

E-MAIL

ESTABELECIMENTO

NOME

LOCALIDADE

MUNICÍPIO UF

TELEFONE ( ) CIDADE + PRÓXIMA

DISTÂNCIA ACESSO RODOVIÁRIO

AFIXO

NOME EXCLUSIVO QUE IDENTIFICARÁ SEUS OVINOS INSCRITOS NO REGISTRO GENEALÓGICO
O AFIXO DEVERÁ TER NO MÁXIMO DUAS PALAVRAS
INDICAR 5 AFIXOS POR ORDEM DE PREFERÊNCIA A FIM DE NÃO COINCIDIR COM OS JÁ EXISTENTES EM NOSSO REGISTRO

RAÇA

1 -

2 -

3 -

4 -

5 -

CASO MENOR DE 18 ANOS

RESPONSÁVEL

ENDEREÇO

MUNICÍPIO UF

CNPJ/CPF

ASS. DO RESPONSÁVEL

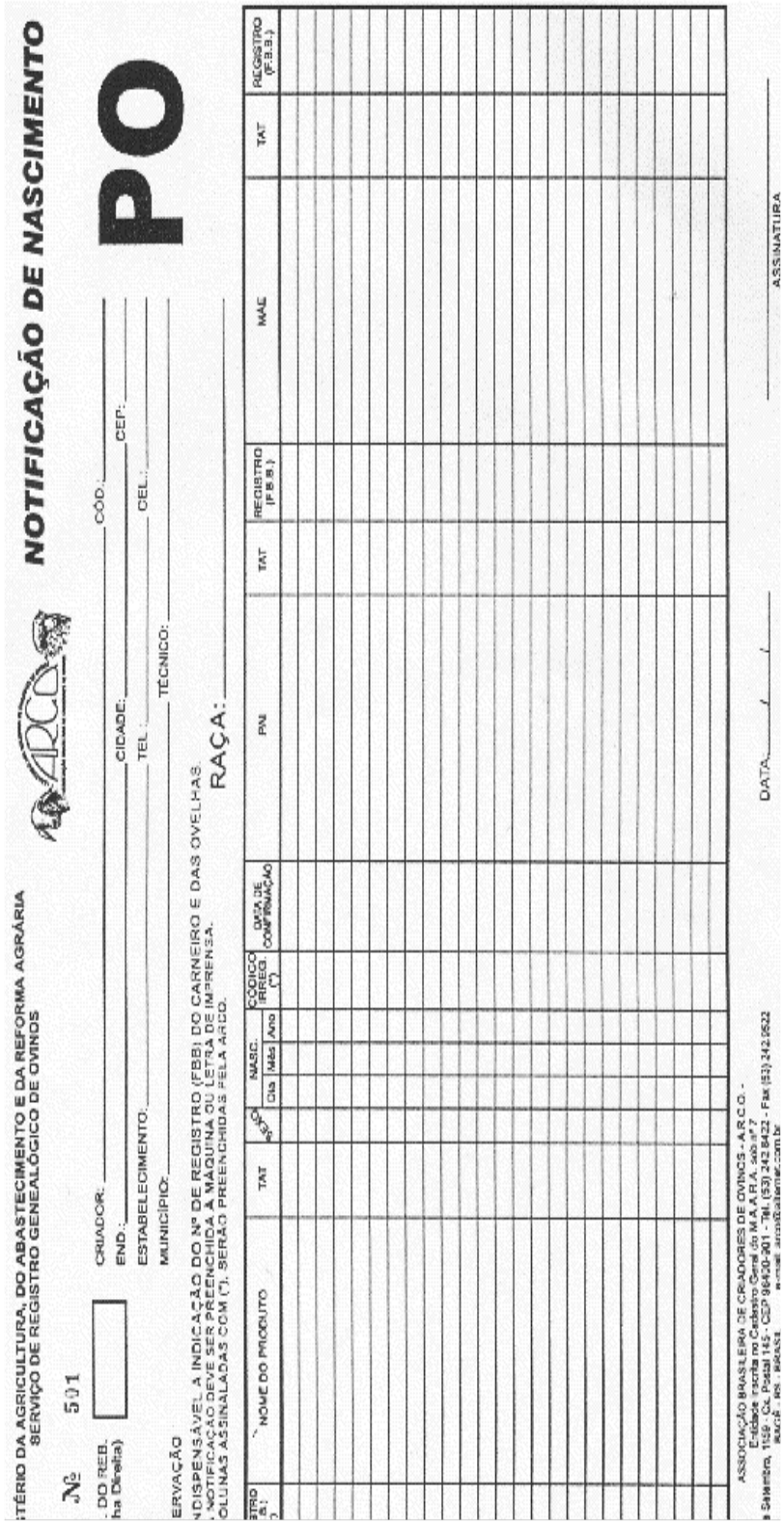
DATA

ASS. DO CRIADOR

Av. 7 de Setembro, 1159 - Cx. Postal 145 - Bagé/RS - CEP 96400-901

Anexo 2

Registro PO



Anexo 3

Registro Base

A.R.C.O.

Ficha de Inspeção do Rebanho Base

ATENÇÃO

Remeta esta via com urgência à ARCO, para a expedição do(s) certificado(s).

Nº 2803

Criador: ..... Data: .....

Est. ..... Munic. .....

Raça: .....

	MACHOS	FÊMEAS	TOTAL
APRESENTADOS			
REGISTRADOS			

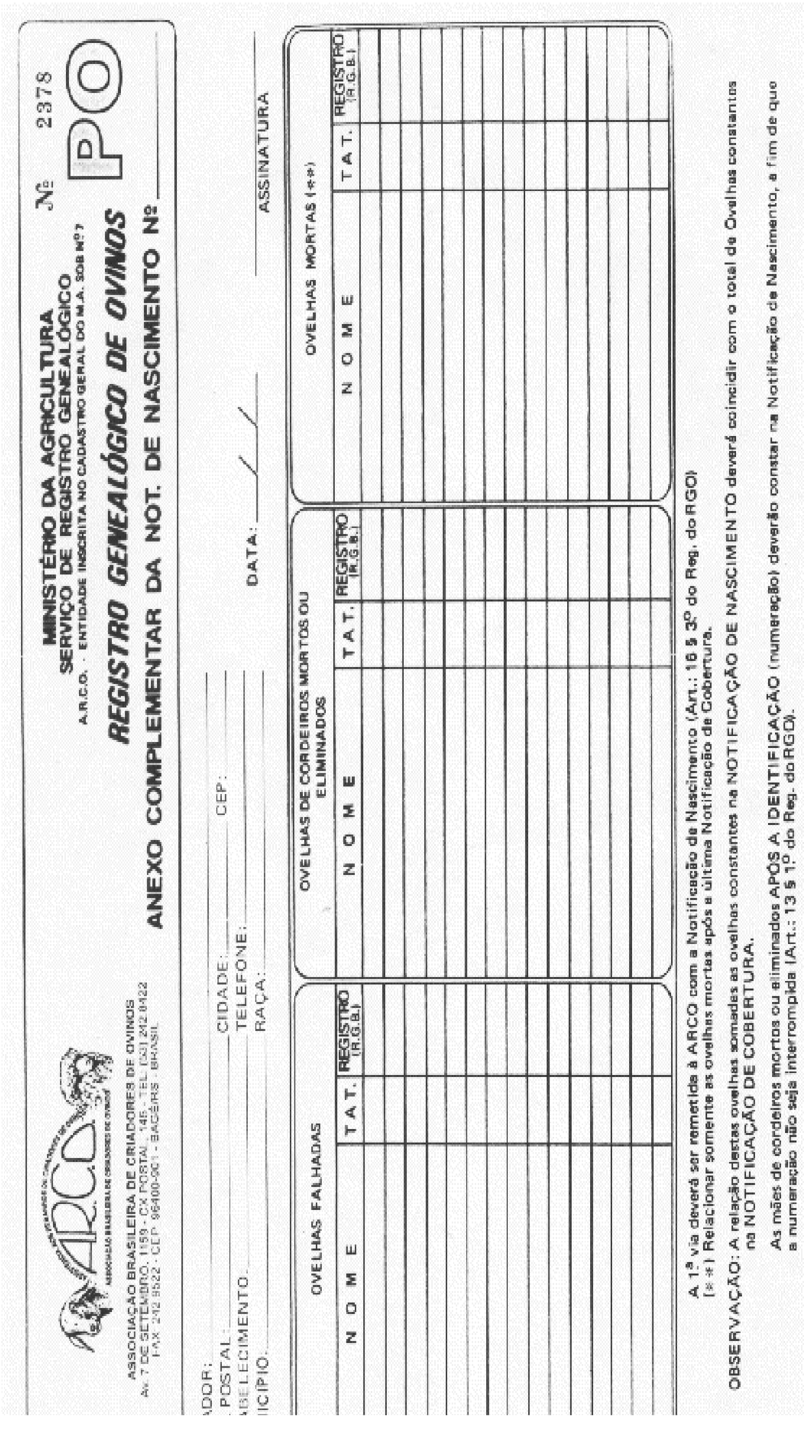
REGISTRADOS

MACHOS						FÊMEAS					
TAT.	NASC.	TAT.	NASC.	TAT.	NASC.	TAT.	NASC.	TAT.	NASC.	TAT.	NASC.

Na raça MORADA NOVA especificar a variedade.

Anexo 4

Registro Complementar



ASSOCIAÇÃO BRASILEIRA DE CRIADORES DE OVINOS
Av. 7 DE SETEMBRO, 1159 - CX POSTAL 145 - TEL (53) 242 8422
FAX 242 8522 - CEP 96400-901 - BAGÉ/RS - BRASIL

MINISTÉRIO DA AGRICULTURA
SERVIÇO DE REGISTRO GENEALÓGICO
A.R.C.O. - ENTIDADE INSCRITA NO CADASTRO GERAL DO M.A. SOB Nº 7

Nº 2378

REGISTRO GENEALÓGICO DE OVINOS

PO

ANEXO COMPLEMENTAR DA NOT. DE NASCIMENTO Nº ____

...DOR: ____
...POSTAL: ____ CIDADE: ____ CEP: ____
...BELECIMENTO: ____ TELEFONE: ____
...ICIPIO: ____ RAÇA: ____ DATA: __/__/__ ____ ASSINATURA

OVELHAS FALHADAS			OVELHAS DE CORDEIROS MORTOS OU ELIMINADOS			OVELHAS MORTAS (**)		
NOME	TAT.	REGISTRO (R.G.B.)	NOME	TAT.	REGISTRO (R.G.B.)	NOME	TAT.	REGISTRO (R.G.B.)

A 1ª via deverá ser remetida à ARCO com a Notificação de Nascimento (Art.: 16 § 3º do Reg. do RGO)
(**) Relacionar somente as ovelhas mortas após a última Notificação de Cobertura.

OBSERVAÇÃO: A relação destas ovelhas somadas as ovelhas constantes na NOTIFICAÇÃO DE NASCIMENTO deverá coincidir com o total de Ovelhas constantes na NOTIFICAÇÃO DE COBERTURA.

As mães de cordeiros mortos ou eliminados APÓS A IDENTIFICAÇÃO (numeração) deverão constar na Notificação de Nascimento, a fim de que a numeração não seja interrompida (Art.: 13 § 1º do Reg. do RGO).

Anexo 5

OvinosPO.xml

```
<?xml version="1.0"?>
<OvinosPO xmlns="x-schema:OvinosPO.biz">
  <Raca>
    <DescricaoRaca>Suffolk</DescricaoRaca>
    <FBB Codigo="O000003">O000120</FBB>
    <Tatuagem>01</Tatuagem>
    <DuplaTatuagem Tat="s"/>
    <GeracoesControladas>1</GeracoesControladas>
    <Confirmado Conf="C"/>
    <Nome>Princesa do Mar 01</Nome>
    <Nacionalidade>Brasileira</Nacionalidade>
    <FBBPai>O000235</FBBPai>
    <RegistroPaiExterior>CH002365</RegistroPaiExterior>
    <NomePai>Chico 53</NomePai>
    <FBBMae>O000065</FBBMae>
    <RegistroMaeExterior>CH000125</RegistroMaeExterior>
    <NomeMae>Favorita 14</NomeMae>
    <DataNascimento>1998-05-12</DataNascimento>
    <Sexo Sex="M"/>
    <CodigoCriador>123456</CodigoCriador>
    <NomeCriador>Joao</NomeCriador>
    <CodigoAfixo>01</CodigoAfixo>
    <CodigoEstabelecimento>10</CodigoEstabelecimento>
    <NomeEstabelecimento>Cordilheira</NomeEstabelecimento>
    <CodigoProprietario>123456</CodigoProprietario>
    <NomeProprietario>Joao</NomeProprietario>
    <NumeroDocumentConfirm>100</NumeroDocumentConfirm>
    <NumeroDocumentNotific>5688</NumeroDocumentNotific>
    <DataRegistro>1998-12-15</DataRegistro>
    <VivoMorto Vivo="N"/>
    <DataMorte>1999-12-25</DataMorte>
    <Transferido Trans="S"/>
    <DataConfirmacao>1998-02-15</DataConfirmacao>
    <DataAnotacaoConfirmacao>1998-02-15</DataAnotacaoConfirmacao>
    <CodigoTecnico>123</CodigoTecnico>
    <NomeTecnico>Francisco</NomeTecnico>
    <DataUltimaTransferencia>1999-10-20</DataUltimaTransferencia>
    <DataAnotacaoUltimaTransf>1999-10-21</DataAnotacaoUltimaTransf>
    <TipoCobertura Tipocob="MN"/>
    <TipoParto Tipopart="2"/>
    <AtaConfirmacao>258</AtaConfirmacao>
    <DataPrimeiroCertificado>1999-10-02</DataPrimeiroCertificado>
    <DataDigitacao>1998-12-20</DataDigitacao>
  </Raca>
  <Raca>
    <DescricaoRaca>Suffolk</DescricaoRaca>
    <FBB Codigo="O00025">O00236</FBB>
    <Tatuagem>26</Tatuagem>
    <DuplaTatuagem Tat="S"/>
    <GeracoesControladas>1</GeracoesControladas>
    <Confirmado Conf="R"/>
    <Nome>Pedrao 26</Nome>
    <Nacionalidade>Brasileiro</Nacionalidade>
    <FBBPai>O00236</FBBPai>
```

```
    <RegistroPaiExterior/>
    <NomePai>Tigrao 56</NomePai>
    <FBBMae>O65894</FBBMae>
    <RegistroMaeExterior/>
    <NomeMae>Chiquita 12</NomeMae>
    <DataNascimento>1999-12-01</DataNascimento>
    <Sexo Sex="F"/>
    <CodigoCriador>456</CodigoCriador>
    <NomeCriador>Abelar</NomeCriador>
    <CodigoAfixo>4</CodigoAfixo>
    <CodigoEstabelecimento>78</CodigoEstabelecimento>
    <NomeEstabelecimento>Corunilha</NomeEstabelecimento>
    <CodigoProprietario>456</CodigoProprietario>
    <NomeProprietario>Paulo</NomeProprietario>
    <NumeroDocumentConfirm>741</NumeroDocumentConfirm>
    <NumeroDocumentNotific>963</NumeroDocumentNotific>
    <DataRegistro>2000-02-15</DataRegistro>
    <VivoMorto Vivo="S"/>
    <DataMorte/>
    <Transferido Trans="N"/>
    <DataConfirmacao>2000-01-20</DataConfirmacao>
    <DataAnotacaoConfirmacao>2000-01-22</DataAnotacaoConfirmacao>
    <CodigoTecnico>13</CodigoTecnico>
    <NomeTecnico>Marcio</NomeTecnico>
    <DataUltimaTransferencia/>
    <DataAnotacaoUltimaTransf/>
    <TipoCobertura Tipocob="IA"/>
    <TipoParto Tipopart="1"/>
    <AtaConfirmacao>159</AtaConfirmacao>
    <DataPrimeiroCertificado>2000-02-10</DataPrimeiroCertificado>
    <DataDigitacao>1999-06-17</DataDigitacao>
  </Raca>
</OvinosPO>
```

Anexo 6

OvinosBA.xml

```
<?xml version="1.0"?>
<OvinosBA xmlns="x-schema:OvinosBA.biz">
  <Raca>
    <DescricaoRaca>Texel</DescricaoRaca>
    <FBB codigo="O3">O00012</FBB>
    <CodigoAfixo>12</CodigoAfixo>
    <Nome>Falcao 20</Nome>
    <Tatuagem>20</Tatuagem>
    <DuplaTatuagem Tat="S"/>
    <Sexo Sex="M"/>
    <AnoNascimento>1998</AnoNascimento>
    <DataConfirmacao>2000-01-15</DataConfirmacao>
    <DataAnotacaoConfirmacao>2000-01-15</DataAnotacaoConfirmacao>
    <CodigoCriador>125</CodigoCriador>
    <NomeCriador>Joao</NomeCriador>
    <CodigoProprietario>125</CodigoProprietario>
    <NomeProprietario>Joao</NomeProprietario>
    <DataRegistro>2000-02-20</DataRegistro>
    <AtaConfirmacao>100</AtaConfirmacao>
    <NumeroDocumentConfirm>12365</NumeroDocumentConfirm>
    <CodigoTecnico>13</CodigoTecnico>
    <NomeTecnico>Pedro</NomeTecnico>
    <CodigoEstabelecimento>147</CodigoEstabelecimento>
    <NomeEstabelecimento>Corunilha</NomeEstabelecimento>
    <Transferido Trans="N"/>
    <DataUltimaTransferencia/>
    <DataAnotacaoUltimaTransf/>
    <DataPrimeiroCertificado>2000-02-25</DataPrimeiroCertificado>
    <VivoMorto Vivo="S"/>
    <DataMorte/>
    <DataDigitacao>2000-08-08</DataDigitacao>
  </Raca>
  <Raca>
    <DescricaoRaca>Texel</DescricaoRaca>
    <FBB codigo="O1">O000010</FBB>
    <CodigoAfixo>125</CodigoAfixo>
    <Nome>Tigrao 15</Nome>
    <Tatuagem>15</Tatuagem>
    <DuplaTatuagem Tat="N"/>
    <Sexo Sex="F"/>
    <AnoNascimento>1984</AnoNascimento>
    <DataConfirmacao>1998-12-10</DataConfirmacao>
    <DataAnotacaoConfirmacao>1998-12-20</DataAnotacaoConfirmacao>
    <CodigoCriador>369</CodigoCriador>
    <NomeCriador>Paulo</NomeCriador>
    <CodigoProprietario>256</CodigoProprietario>
    <NomeProprietario>Jose</NomeProprietario>
    <DataRegistro>1999-01-10</DataRegistro>
    <AtaConfirmacao>456</AtaConfirmacao>
    <NumeroDocumentConfirm>4569</NumeroDocumentConfirm>
    <CodigoTecnico>16</CodigoTecnico>
    <NomeTecnico>Francisco</NomeTecnico>
    <CodigoEstabelecimento>456</CodigoEstabelecimento>
    <NomeEstabelecimento>Castelo</NomeEstabelecimento>
```

```
    <Transferido Trans="S"/>
    <DataUltimaTransferencia>2000-02-25</DataUltimaTransferencia>
    <DataAnotacaoUltimaTransf>2000-02-25</DataAnotacaoUltimaTransf>
    <DataPrimeiroCertificado>1998-12-30</DataPrimeiroCertificado>
    <VivoMorto Vivo="N"/>
    <DataMorte>2000-04-15</DataMorte>
    <DataDigitacao>2000-08-08</DataDigitacao>
  </Raca>
  <Raca>
    <DescricaoRaca>Texel</DescricaoRaca>
    <FBB codigo="O2">O00125</FBB>
    <CodigoAfixo>459</CodigoAfixo>
    <Nome>Rapadura 35</Nome>
    <Tatuagem>35</Tatuagem>
    <DuplaTatuagem Tat="S"/>
    <Sexo Sex="M"/>
    <AnoNascimento>1988</AnoNascimento>
    <DataConfirmacao>1995-10-15</DataConfirmacao>
    <DataAnotacaoConfirmacao>1995-10-15</DataAnotacaoConfirmacao>
    <CodigoCriador>125</CodigoCriador>
    <NomeCriador>Abelar</NomeCriador>
    <CodigoProprietario>896</CodigoProprietario>
    <NomeProprietario>Abelar</NomeProprietario>
    <DataRegistro>1996-01-15</DataRegistro>
    <AtaConfirmacao>856</AtaConfirmacao>
    <NumeroDocumentConfirm>7415</NumeroDocumentConfirm>
    <CodigoTecnico>17</CodigoTecnico>
    <NomeTecnico>Perello</NomeTecnico>
    <CodigoEstabelecimento>369</CodigoEstabelecimento>
    <NomeEstabelecimento>Vila Nova</NomeEstabelecimento>
    <Transferido Trans="S"/>
    <DataUltimaTransferencia>2000-01-10</DataUltimaTransferencia>
    <DataAnotacaoUltimaTransf>2000-01-10</DataAnotacaoUltimaTransf>
    <DataPrimeiroCertificado>1996-12-20</DataPrimeiroCertificado>
    <VivoMorto Vivo="N"/>
    <DataMorte>2000-05-15</DataMorte>
    <DataDigitacao>2000-08-08</DataDigitacao>
  </Raca>
</OvinosBA>
```

Anexo 7

Fichario.xml

```
<?xml version="1.0"?>
<ARCO xmlns="x-schema:Fichario.biz">
  <Fichario>
    <Socio Codigo="S1">1</Socio>
    <Nome>Joao</Nome>
    <Endereco>20 de setembro, 1720</Endereco>
    <Bairro>centro</Bairro>
    <CXP>13</CXP>
    <CEP>96400</CEP>
    <Localidade>Bage</Localidade>
    <Cidade>Bage</Cidade>
    <Estado>RS</Estado>
    <DDD>053</DDD>
    <Fone>421-3659</Fone>
    <Ramal/>
    <Fax/>
    <CPF>1236589</CPF>
    <DataNasc>1980-06-15</DataNasc>
    <Profissao>Professor</Profissao>
    <Naturalidade>Pelotense</Naturalidade>
    <Sexo Sex="M"/>
    <Representante>Alguem</Representante>
    <DataMatric>1999-10-20</DataMatric>
    <Responsavel/>
    <TipoSocio Tiposoc="C"/>
    <TipoPessoa Tipopes="F"/>
    <CodigoTecnico>13</CodigoTecnico>
    <Email>jamc@urcamp.tche.br</Email>
  </Fichario>
  <Fichario>
    <Socio Codigo="S2">2</Socio>
    <Nome>Pedro</Nome>
    <Endereco>Barao do Triunfo, 124</Endereco>
    <Bairro>Tiaraju</Bairro>
    <CXP/>
    <CEP>96400</CEP>
    <Localidade>Bage</Localidade>
    <Cidade>Bage</Cidade>
    <Estado>RS</Estado>
    <DDD>053</DDD>
    <Fone/>
    <Ramal/>
    <Fax/>
    <CPF>265987</CPF>
    <DataNasc>1986-12-14</DataNasc>
    <Profissao>Pedreiro</Profissao>
    <Naturalidade>Cacimbinhas</Naturalidade>
    <Sexo Sex="M"/>
    <Representante>Pedrao</Representante>
    <DataMatric>2000-06-04</DataMatric>
    <Responsavel>Tigrao</Responsavel>
    <TipoSocio Tiposoc="R"/>
    <TipoPessoa Tipopes="J"/>
    <CodigoTecnico>15</CodigoTecnico>
```

```
    <Email/>
  </Fichario>
  <Fichario>
    <Socio Codigo="S3">3</Socio>
    <Nome>Maria</Nome>
    <Endereco>Pery Coronel, 171</Endereco>
    <Bairro>Laranjeiras</Bairro>
    <CXP>17</CXP>
    <CEP>96400-051</CEP>
    <Localidade>Zona Rural</Localidade>
    <Cidade>Dom Pedrito</Cidade>
    <Estado>RS</Estado>
    <DDD>056</DDD>
    <Fone>256-8965</Fone>
    <Ramal>123</Ramal>
    <Fax>256-6523</Fax>
    <CPF>456987</CPF>
    <DataNasc>1950-02-20</DataNasc>
    <Profissao>Medico</Profissao>
    <Naturalidade>Acegua</Naturalidade>
    <Sexo Sex="F"/>
    <Representante>Faustao</Representante>
    <DataMatric>2000-06-18</DataMatric>
    <Responsavel>Fabricio</Responsavel>
    <TipoSocio Tiposoc="C"/>
    <TipoPessoa Tipopes="F"/>
    <CodigoTecnico>13</CodigoTecnico>
    <Email>Maria@alternet.com.br</Email>
  </Fichario>
</ARCO>
```

Anexo 8

OvinosPO.biz

```
<?xml version ="1.0"?>
<!--Generated by XML Authority. Conforms to XML Data subset for IE 5-->
<Schema name = "OvinosPO.biz"
        xmlns = "urn:schemas-microsoft-com:xml-data"
        xmlns:dt = "urn:schemas-microsoft-com:datatypes">
```

Cabeçalho

1ª Etapa

```
<ElementType name = "OvinosPO" content = "eltOnly" order = "seq">
        <element type = "Raca" minOccurs = "0" maxOccurs = "*"/>
</ElementType>

<ElementType name = "Raca" content = "eltOnly" order = "seq">
        <element type = "DescricaoRaca"/>
        <element type = "FBB"/>
        <element type = "Tatuagem"/>
        <element type = "DuplaTatuagem"/>
        <element type = "GeracoesControladas" minOccurs = "0" maxOccurs = "1"/>
        <element type = "Confirmado"/>
        <element type = "Nome"/>
        <element type = "Nacionalidade" minOccurs = "0" maxOccurs = "1"/>
        <element type = "FBBPai"/>
        <element type = "RegistroPaiExterior" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomePai"/>
        <element type = "FBBMae"/>
        <element type = "RegistroMaeExterior" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomeMae"/>
```

```
        <element type = "DataNascimento"/>
        <element type = "Sexo"/>
        <element type = "CodigoCriador" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomeCriador"/>
        <element type = "CodigoAfixo" minOccurs = "0" maxOccurs = "1"/>
        <element type = "CodigoEstabelecimento" minOccurs = "0" maxOccurs = "*"/>
        <element type = "NomeEstabelecimento" minOccurs = "0" maxOccurs = "*"/>
        <element type = "CodigoProprietario" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomeProprietario" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NumeroDocumentConfirm" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NumeroDocumentNotific" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataRegistro"/>
        <element type = "VivoMorto"/>
        <element type = "DataMorte" minOccurs = "0" maxOccurs = "1"/>
        <element type = "Transferido"/>
        <element type = "DataConfirmacao" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataAnotacaoConfirmacao" minOccurs = "0" maxOccurs = "1"/>
        <element type = "CodigoTecnico" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomeTecnico" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataUltimaTransferencia" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataAnotacaoUltimaTransf" minOccurs = "0" maxOccurs = "1"/>
        <element type = "TipoCobertura"/>
        <element type = "TipoParto"/>
        <element type = "AtaConfirmacao" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataPrimeiroCertificado" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataDigitacao"/>
</ElementType>
```

2ª Etapa

```
<ElementType name = "FBB" content = "textOnly" dt:type = "string">
```

```
            <AttributeType name = "Codigo" dt:type = "id" required = "yes"/>
            <attribute type = "Codigo"/>
        </ElementType>

        <ElementType name = "Tatuagem" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DuplaTatuagem" content = "textOnly" dt:type = "string">
            <AttributeType name = "Tat" dt:type = "enumeration" dt:values = "S N s n" default =
"N"/>
            <attribute type = "Tat"/>
        </ElementType>

        <ElementType name = "GeracoesControladas" content = "textOnly" dt:type = "string"/>
        <ElementType name = "Confirmado" content = "textOnly" dt:type = "string">
            <AttributeType name = "Conf" dt:type = "enumeration" dt:values = "C N R A I c n r a
i" default = "N"/>
            <attribute type = "Conf"/>
        </ElementType>

        <ElementType name = "Nome" content = "textOnly" dt:type = "string"/>
        <ElementType name = "Nacionalidade" content = "textOnly" dt:type = "string"/>
        <ElementType name = "FBBPai" content = "textOnly" dt:type = "string"/>
        <ElementType name = "RegistroPaiExterior" content = "textOnly" dt:type = "string"/>
        <ElementType name = "NomePai" content = "textOnly" dt:type = "string"/>
        <ElementType name = "FBBMae" content = "textOnly" dt:type = "string"/>
        <ElementType name = "RegistroMaeExterior" content = "textOnly" dt:type = "string"/>
        <ElementType name = "NomeMae" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DataNascimento" content = "textOnly" dt:type = "date"/>
        <ElementType name = "Sexo" content = "textOnly" dt:type = "string">
            <AttributeType name = "Sex" dt:type = "enumeration" dt:values = "M F m f"/>
            <attribute type = "Sex"/>
        </ElementType>
```

```
<ElementType name = "CodigoCriador" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeCriador" content = "textOnly" dt:type = "string"/>
<ElementType name = "CodigoAfixo" content = "textOnly" dt:type = "int"/>
<ElementType name = "CodigoEstabelecimento" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeEstabelecimento" content = "textOnly" dt:type = "string"/>
<ElementType name = "CodigoProprietario" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeProprietario" content = "textOnly" dt:type = "string"/>
<ElementType name = "NumeroDocumentConfirm" content = "textOnly" dt:type = "string"/>
<ElementType name = "NumeroDocumentNotific" content = "textOnly" dt:type = "string"/>
<ElementType name = "DataRegistro" content = "textOnly" dt:type = "date"/>
<ElementType name = "VivoMorto" content = "textOnly" dt:type = "string">
        <AttributeType name = "Vivo" dt:type = "enumeration" dt:values = "S N s n" default =
"S"/>
        <attribute type = "Vivo"/>
</ElementType>

<ElementType name = "DataMorte" content = "textOnly" dt:type = "date"/>
<ElementType name = "Transferido" content = "textOnly" dt:type = "string">
        <AttributeType name = "Trans" dt:type = "enumeration" dt:values = "S N s n" default =
"N"/>
        <attribute type = "Trans"/>
</ElementType>

<ElementType name = "DataConfirmacao" content = "textOnly" dt:type = "date"/>
<ElementType name = "DataAnotacaoConfirmacao" content = "textOnly" dt:type = "date"/>
<ElementType name = "CodigoTecnico" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeTecnico" content = "textOnly" dt:type = "string"/>
<ElementType name = "DataUltimaTransferencia" content = "textOnly" dt:type = "date"/>
<ElementType name = "DataAnotacaoUltimaTransf" content = "textOnly" dt:type = "date"/>
<ElementType name = "TipoCobertura" content = "textOnly" dt:type = "string">
```

.

```
                <AttributeType name = "Tipocob" dt:type = "enumeration" dt:values = "MN IA TE mn
ia te" default = "MN"/>

                <attribute type = "Tipocob"/>

        </ElementType>


        <ElementType name = "TipoParto" content = "textOnly" dt:type = "string">

                <AttributeType name = "Tipopart" dt:type = "enumeration" dt:values = "1 2 3 4 5"
default = "1"/>

                <attribute type = "Tipopart"/>

        </ElementType>


        <ElementType name = "AtaConfirmacao" content = "textOnly" dt:type = "string"/>

        <ElementType name = "DataPrimeiroCertificado" content = "textOnly" dt:type = "date"/>

        <ElementType name = "DescricaoRaca" content = "textOnly" dt:type = "string"/>

        <ElementType name = "DataDigitacao" content = "textOnly" dt:type = "date"/>

</Schema>
```

Anexo 9

OvinosBA.biz

```
<?xml version ="1.0"?>
<!--Generated by XML Authority. Conforms to XML Data subset for IE 5-->
<Schema name = "OvinosBA.biz"
        xmlns = "urn:schemas-microsoft-com:xml-data"
        xmlns:dt = "urn:schemas-microsoft-com:datatypes">
        <ElementType name = "OvinosBA" content = "eltOnly" order = "seq">
                <element type = "Raca" minOccurs = "0" maxOccurs = "*"/>
        </ElementType>

        <ElementType name = "Raca" content = "eltOnly" order = "seq">
                <element type = "Descricao"/>
                <element type = "FBB"/>
                <element type = "CodigoAfixo" minOccurs = "0" maxOccurs = "1"/>
                <element type = "Nome"/>
                <element type = "Tatuagem"/>
                <element type = "DuplaTatuagem"/>
                <element type = "Sexo"/>
                <element type = "AnoNascimento"/>
                <element type = "DataConfirmacao"/>
                <element type = "DataAnotacaoConfirmacao" minOccurs = "0" maxOccurs = "1"/>
                <element type = "CodigoCriador" minOccurs = "0" maxOccurs = "1"/>
                <element type = "NomeCriador"/>
                <element type = "CodigoProprietario" minOccurs = "0" maxOccurs = "1"/>
                <element type = "NomeProprietario" minOccurs = "0" maxOccurs = "1"/>
                <element type = "DataRegistro"/>
                <element type = "AtaConfirmacao" minOccurs = "0" maxOccurs = "1"/>
                <element type = "NumeroDocumentConfirm" minOccurs = "0" maxOccurs = "1"/>
```

```
        <element type = "CodigoTecnico" minOccurs = "0" maxOccurs = "1"/>
        <element type = "NomeTecnico" minOccurs = "0" maxOccurs = "1"/>
        <element type = "CodigoEstabelecimento" minOccurs = "0" maxOccurs = "*"/>
        <element type = "NomeEstabelecimento" minOccurs = "0" maxOccurs = "*"/>
        <element type = "Transferido"/>
        <element type = "DataUltimaTransferencia" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataAnotacaoUltimaTransf" minOccurs = "0" maxOccurs = "1"/>
        <element type = "DataPrimeiroCertificado" minOccurs = "0" maxOccurs = "1"/>
        <element type = "VivoMorto"/>
        <element type = "DataMorte" minOccurs = "0" maxOccurs = "1"/>
</ElementType>

<ElementType name = "FBB" content = "textOnly" dt:type = "string">
        <AttributeType name = "codigo" dt:type = "id"/>
        <attribute type = "codigo"/>
</ElementType>

<ElementType name = "CodigoAfixo" content = "textOnly" dt:type = "int"/>
<ElementType name = "Nome" content = "textOnly" dt:type = "string"/>
<ElementType name = "Tatuagem" content = "textOnly" dt:type = "string"/>
<ElementType name = "DuplaTatuagem" content = "textOnly" dt:type = "string"/>
<ElementType name = "Sexo" content = "textOnly" dt:type = "string"/>
<ElementType name = "AnoNascimento" content = "textOnly" dt:type = "string"/>
<ElementType name = "DataConfirmacao" content = "textOnly" dt:type = "date"/>
<ElementType name = "DataAnotacaoConfirmacao" content = "textOnly" dt:type = "date"/>
<ElementType name = "CodigoCriador" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeCriador" content = "textOnly" dt:type = "string"/>
<ElementType name = "CodigoProprietario" content = "textOnly" dt:type = "int"/>
<ElementType name = "NomeProprietario" content = "textOnly" dt:type = "string"/>
<ElementType name = "DataRegistro" content = "textOnly" dt:type = "date"/>
```

```
        <ElementType name = "AtaConfirmacao" content = "textOnly" dt:type = "string"/>
        <ElementType name = "NumeroDocumentConfirm" content = "textOnly" dt:type = "string"/>
        <ElementType name = "CodigoTecnico" content = "textOnly" dt:type = "int"/>
        <ElementType name = "NomeTecnico" content = "textOnly" dt:type = "string"/>
        <ElementType name = "CodigoEstabelecimento" content = "textOnly" dt:type = "int"/>
        <ElementType name = "NomeEstabelecimento" content = "textOnly" dt:type = "string"/>
        <ElementType name = "Transferido" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DataUltimaTransferencia" content = "textOnly" dt:type = "date"/>
        <ElementType name = "DataAnotacaoUltimaTransf" content = "textOnly" dt:type = "date"/>
        <ElementType name = "DataPrimeiroCertificado" content = "textOnly" dt:type = "date"/>
        <ElementType name = "VivoMorto" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DataMorte" content = "textOnly" dt:type = "date"/>
        <ElementType name = "Descricao" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DataDigitacao" content = "textOnly" dt:type = "date"/>
</Schema>
```

Anexo 10

Fichario.biz

```
<?xml version ="1.0"?>
<!--Generated by XML Authority. Conforms to XML Data subset for IE 5-->
<Schema name = "Fichario.biz"
        xmlns = "urn:schemas-microsoft-com:xml-data"
        xmlns:dt = "urn:schemas-microsoft-com:datatypes">
       <ElementType name = "Fichario" content = "eltOnly" order = "seq">
              <element type = "Socio"/>
              <element type = "Nome"/>
              <element type = "Endereco"/>
              <element type = "Bairro" minOccurs = "0" maxOccurs = "1"/>
              <element type = "CXP" minOccurs = "0" maxOccurs = "1"/>
              <element type = "CEP"/>
              <element type = "Localidade"/>
              <element type = "Cidade"/>
              <element type = "Estado"/>
              <element type = "DDD" minOccurs = "1" maxOccurs = "*"/>
              <element type = "Fone" minOccurs = "0" maxOccurs = "*"/>
              <element type = "Ramal" minOccurs = "0" maxOccurs = "*"/>
              <element type = "Fax" minOccurs = "0" maxOccurs = "*"/>
              <element type = "CPF"/>
              <element type = "DataNasc"/>
              <element type = "Profissao" minOccurs = "0" maxOccurs = "*"/>
              <element type = "Naturalidade"/>
              <element type = "Sexo"/>
              <element type = "Representante" minOccurs = "0" maxOccurs = "*"/>
              <element type = "DataMatric"/>
              <element type = "Responsavel" minOccurs = "0" maxOccurs = "*"/>
```

```
        <element type = "TipoSocio"/>
        <element type = "TipoPessoa"/>
        <element type = "CodigoTecnico" minOccurs = "1" maxOccurs = "*"/>
        <element type = "Email" minOccurs = "0" maxOccurs = "*"/>
</ElementType>

<ElementType name = "Socio" content = "textOnly" dt:type = "string">
        <AttributeType name = "Codigo" dt:type = "id"/>
        <attribute type = "Codigo"/>
</ElementType>

<ElementType name = "Nome" content = "textOnly" dt:type = "string"/>
<ElementType name = "Endereco" content = "textOnly" dt:type = "string"/>
<ElementType name = "Bairro" content = "textOnly" dt:type = "string"/>
<ElementType name = "CXP" content = "textOnly" dt:type = "string"/>
<ElementType name = "CEP" content = "textOnly" dt:type = "string"/>
<ElementType name = "Localidade" content = "textOnly" dt:type = "string"/>
<ElementType name = "Cidade" content = "textOnly" dt:type = "string"/>
<ElementType name = "Estado" content = "textOnly" dt:type = "string"/>
<ElementType name = "DDD" content = "textOnly" dt:type = "string"/>
<ElementType name = "Fone" content = "textOnly" dt:type = "string"/>
<ElementType name = "Ramal" content = "textOnly" dt:type = "string"/>
<ElementType name = "Fax" content = "textOnly" dt:type = "string"/>
<ElementType name = "CPF" content = "textOnly" dt:type = "string"/>
<ElementType name = "DataNasc" content = "textOnly" dt:type = "date"/>
<ElementType name = "Profissao" content = "textOnly" dt:type = "string"/>
<ElementType name = "Naturalidade" content = "textOnly" dt:type = "string"/>
<ElementType name = "Sexo" content = "textOnly" dt:type = "string">
        <AttributeType name = "Sex" dt:type = "enumeration" dt:values = "M F m f"/>
        <attribute type = "Sex"/>
```

```
        </ElementType>

        <ElementType name = "Representante" content = "textOnly" dt:type = "string"/>
        <ElementType name = "DataMatric" content = "textOnly" dt:type = "date"/>
        <ElementType name = "Responsavel" content = "textOnly" dt:type = "string"/>
        <ElementType name = "TipoSocio" content = "textOnly" dt:type = "string">
                <AttributeType name = "Tiposoc" dt:type = "enumeration" dt:values = "C R c r"/>
                <attribute type = "Tiposoc"/>
        </ElementType>

        <ElementType name = "TipoPessoa" content = "textOnly" dt:type = "string">
                <AttributeType name = "Tipopes" dt:type = "enumeration" dt:values = "F J f j"/>
                <attribute type = "Tipopes"/>
        </ElementType>

        <ElementType name = "CodigoTecnico" content = "textOnly" dt:type = "int"/>
        <ElementType name = "Email" content = "textOnly" dt:type = "string"/>
        <ElementType name = "ARCO" content = "eltOnly" order = "seq">
                <element type = "Fichario" minOccurs = "0" maxOccurs = "*"/>
        </ElementType>

</Schema>
```

Anexo 11

AnimaPR.dbf

A A.R.C.O. utiliza um banco de dados em formato Dbase (dbf), sendo o mesmo gerenciado por softwares utilizando a linguagem Clipper.

Estrutura do arquivo de animais registrados Puros de Origem (PO)

ANIMA_PR.dbf

AP_FBB	caracter tamanho 7	Código do ovino
AP_TAT	caracter tamanho 7	Tatuagem do ovino
AP_DUP	caracter tamanho 1	Dupla tatuagem
AP_GER	caracter tamanho 1	Número de gerações controladas
AP_FCN	caracter tamanho 1	Confirmação do ovino
AP_NOM	caracter tamanho 40	Nome do ovino
AP_RAL	caracter tamanho 9	Nacionalidade do ovino
AP_PAI	caracter tamanho 7	Código do pai
AP_PRA	caracter tamanho 9	Registro do pai no exterior
AP_MAE	caracter tamanho 7	Código da mãe
AP_MRA	caracter tamanho 9	Registro da mãe no exterior
AP_DNS	data	Data de nascimento
AP_SEX	caracter tamanho 1	Sexo
AP_CRI	numérico tamanho 5	Código do Criador
AP_AFI	numérico tamanho 5	Código do Afixo
AP_EST	numérico tamanho 5	Código do Estabelecimento
AP_PRO	numérico tamanho 5	Código do Proprietário
AP_NDC	caracter tamanho 6	Número do documento de confirmação
AP_NDN	caracter tamanho 6	Número do documento de notificação
AP_DRG	data	Data de registro
AP_FVI	caracter tamanho 1	Vivo ou morto
AP_DMT	data	Data da morte
AP_FTR	caracter tamanho 1	Transferência
AP_DCO	data	Data da confirmação
AP_DAN	data	Data da anotação da confirmação
AP_TEC	numérico tamanho 3	Código do Técnico
AP_DUT	data	Data da última transferência
AP_DAT	data	Data anotação última transferência
AP_TCO	caracter tamanho 2	Tipo de cobertura
AP_TPA	caracter tamanho 1	Tipo de parto
AP_ATA	caracter tamanho 6	Número da ata de confirmação
AP_DEX	data	Data da 1º expedição de certificado
AP_DUA	data	Data da última atualização

Anexo 12

AnimaBA.dbf

A A.R.C.O. utiliza um banco de dados em formato DBase (dbf), sendo o mesmo gerenciado por softwares utilizando a linguagem Clipper.

Estrutura do arquivo de animais Base (BA)

ANIMA_BA.dbf

AB_FBB	caracter	tamanho 7	Código do ovino
AB_AFI	numérico	tamanho 5	Código do Afixo
AB_NOM	caracter	tamanho 40	Nome do ovino
AB_TAT	caracter	tamanho 7	Tatuagem
AB_DUP	caracter	tamanho 1	Dupla tatuagem
AB_SEX	caracter	tamanho 1	Sexo
AB_DNS	caracter	tamanho 4	Ano de nascimento
AB_DCO	data		Data da confirmação
AB_DAN	data		Data da anotação da confirmação
AB_CRI	numérico	tamanho 5	Código do Criador
AB_PRO	numérico	tamanho 5	Código do Proprietário
AB_DRG	data		Data de registro
AB_ATA	caracter	tamanho 6	Número da ata de confirmação
AB_NDC	caracter	tamanho 6	Número do doc. de confirmação
AB_TEC	numérico	tamanho 3	Código do Técnico
AB_EST	numérico	tamanho 5	Código do Estabelecimento
AB_FTR	caracter	tamanho 1	Transferência
AB_DUT	data		Data da última transferência
AB_DAT	data		Data anotação última transferência
AB_DUA	data		Data da última atualização
AB_DEX	data		Data da expedição do 1º certificado
AB_FVI	caracter	tamanho 1	Vivo ou morto
AB_DMT	data		Data da morte

Anexo 13

Fichario.dbf

A A.R.C.O. utiliza um banco de dados em formato DBase (dbf), sendo o mesmo gerenciado por softwares utilizando a linguagem Clipper.

Estrutura do arquivo de Sócios

FICHARIO.dbf

FI_COD	numérico	tamanho	5	Código do Associado
FI_NOM	caracter	tamanho	40	Nome do Associado
FI_END	caracter	tamanho	40	Endereço
FI_BAI	caracter	tamanho	15	Bairro
FI_CXP	caracter	tamanho	5	Caixa Postal
FI_CEP	caracter	tamanho	9	CEP
FI_CLO	caracter	tamanho	3	Código da Localidade
FI_LOC	caracter	tamanho	25	Nome da Localidade
FI_CID	caracter	tamanho	25	Cidade
FI_EST	caracter	tamanho	2	Estado
FI_DDD	caracter	tamanho	5	DDD
FI_FON	caracter	tamanho	9	Fone
FI_RAL	caracter	tamanho	4	Ramal
FI_FAX	caracter	tamanho	11	Fax
FI_CPF	caracter	tamanho	18	CPF
FI_DNS	caracter	tamanho	8	Data de nascimento
FI_PRF	caracter	tamanho	15	Profissão
FI_NAT	caracter	tamanho	20	Naturalidade
FI_SEX	caracter	tamanho	1	Sexo
FI_REP	caracter	tamanho	30	Representante junto a A.R.C.O.
FI_DMT	data			Data da matrícula
FI_RES	numérico	tamanho	5	Código do responsável
FI_FSC	caracter	tamanho	1	Tipo de Sócio
FI_FPS	caracter	tamanho	1	Tipo de Pessoa (física ou jurídica)
FI_CTE	numérico	tamanho	3	Código do Técnico
FI_DUA	data			Data da última atualização

Bibliografia

[ALL 2000] ALSHULER, Liora . **Schema Repositories** . Disponível em: <http://xml.com/ pub/2000/01/26/feature/index.html>. Acesso em: jan. 2000.

[ANL 98] ANDREW, Layman. **XML Data**. Disponível em:< http://www.w3.org/tr /1998/note-xml-data-0105/>.Acesso em: jan. 1998.

[ARL 99] ARSHOL, Lars Marius. **Introduction to XML**. Disponível em: <http://www. stud.ifi.uio.no/~imariusg/download/xml/xml-eng.html>. Acesso em: ago. 1999.

[ASM 99] ASHOK, Malhotra. **XML Schema Requirements**. Disponível em: <http://www.w3.org/tr/note-xml-schema-req>. Acesso em: fev. 1999.

[BOJ 97] BOSAK, J. **XML Java and the Future of the Web XML Principles, Tools and Techniques.** Disponível em: <http://xml.com/pub/ 97/20/indice.html>. Acesso em: set. 1997.

[BRD 97] BRAY, T. ; DEROSE, S.Extensible Markup Language (XML) Part 2: Linking. **XML Principles, Tools and Techniques** ,[S.l.], v.2,n.4,p.219-227, Sept. 1997.

[BRT 98] BRAY, Tim. **Extensible Markup Language (XML) 1.0**. Disponível em:<http://wwwest.uniandes.edu.co/~l-arcini/spec.html>. Acesso em: fev. 1998.

[COR 2000] COVER, Robin. **The XML Cover Page**. Disponível em: <http://xml.com/pub/ coverpage.html#ni2000-02-04-a>. Acesso em: fev. 2000.

[HAP 99] HALL, Prentice. **XML Aplicações Práticas**. Rio de Janeiro: Campus, 1997.

[HEC 98] HEUSER, C. **Projeto de Banco de Dados**. Porto Alegre: Sagra Luzzatto, 1998.

[HOD 99] HOLLANDER, Dave. **Namespaces in XML**. Disponível em: <http:// www.w3.org/ tr/1999/rec-xml-names-19990114>. Acesso em: jan. 1999.

[LIQ 99] LILLEY,C. ; QUINT, V.**Extensible Stylesheet Language (XSL)**. Disponível em: < http://www.w3.org/Style/XSL/>. Acesso em: nov. 1999.

[LIR 97] LIGHT, Richard. **Iniciando em XML**. São Paulo: Makron Books, 1997

[MAD 2000] MARTIN, Didier. **A Class Act**. Disponível em:http://www.xml.com/ pub/2000/02/02/style/index.html>. Acesso em: fev. 2000.

[OGU 99] OGBUJI, Uche. **XML´s not HTML**. Disponível em: <http:// www.linuxworld.com/linuxworld/lw-03-xml.html>. Acesso em: nov. 1999.

[PES 99] PEPPER, Steve. **The Whirlwind Guide to SGML & XML**. Disponível em: <http://www.infotek.no/sgmltool/guide.htm>. Acesso em: dez. 1999.

[STJ 99] STEVEN, J. de Rose. **XML Xlink Requirements**. Disponível em: <http://www.w3.org/tr/note-xlink-req/>. Acesso em: fev. 1999.

[WAN 98] WALSH, Norman. **A Techinal Introduction to XML.** Disponível em: <http://xml.com/pub/98/10/guide0.html>. Acesso em: out. 1998.

[WOR 2000] WORDEN, Robert. **XML E-Business Standards**. Disponível em:<http://xml.com/pub/2000/01/ebusiness/index.html>. Acesso em: jan. 2000.