UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA

COMPUTAÇÃO

Fernando Carlos Ponce de Leon Antunes

UMA LINGUAGEM DE DEFINIÇÃO DA CORRELAÇÃO ENTRE METADADOS DE BIBLIOTECAS DIGITAIS PROPRIETÁRIAS EM METADADOS DUBLIN CORE E SEU USO EM SERVIDORES Z39.50

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Roberto Willrich

Florianópolis, Fevereiro de 2005

UMA LINGUAGEM DE DEFINIÇÃO DA CORRELAÇÃO ENTRE METADADOS DE BIBLIOTECAS DIGITAIS PROPRIETÁRIAS EM METADADOS DUBLIN CORE E SEU USO EM SERVIDORES Z39.50

Fernando Carlos Ponce de Leon Antunes

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Prof. Dr. Raul Sidney Wazlawick
Coordenador do Curso

Banca Examinadora

Prof. Dr. Roberto Willrich (Orientador)
INE/UFSC

Prof. Dr. Vitório Bruno Mazzola
INE/UFSC

Prof. Dr. Pierre de Saqui-Sannes
ENSICA/França

Prof. Dr. Luiz Fernando Rust da Costa Carmo
NCE/UFRJ

Agradecimentos

Agradeço primeiramente a Deus, que ao longo da minha vida vem fazendo incontáveis concessões, dentre as quais a determinação pessoal de concluir este trabalho. Agradeço também, a minha esposa Patrizia e ao meu filho Rafael, que são a razão pela qual eu prossigo na minha caminhada. Ao professor Roberto Willrich, um agradecimento especial pela sua atenção, competência e profissionalismo dispensados a tarefa de me orientar na realização deste trabalho.

LISTA DE FIGURAS

RESUMO

O número de organizações que criam suas próprias bibliotecas digitais vem crescendo. Isto, pois as bibliotecas digitais oferecem um modo eficiente de organização no armazenamento do acervo e uma facilidade de recuperação de informação. Todo o conhecimento que está armazenado nas bibliotecas digitais tem seu valor reconhecido à medida que se torna disponível à pesquisa. Desta forma, a interoperabilidade entre as bibliotecas digitais representa uma importante questão a ser resolvida. Algumas iniciativas tentam resolver este problema, como o protocolo Z39.50 e o padrão de metadados Dublin Core. Mas existem diversas implementações de bibliotecas digitais proprietárias que não adotam tais iniciativas. Hoje existem soluções permitindo adaptar padrões de interoperabilidade às bibliotecas proprietárias. Mas a instalação de tais soluções é complexa, exigindo até a recodificação. O objetivo desta dissertação é propor uma solução para facilitar a adaptação de servidores Z39.50 a bibliotecas digitais proprietárias que utilizam banco de dados relacional. Para tal, foi definida uma linguagem baseada em XML para a definição da correlação entre metadados de bibliotecas digitais proprietárias em metadados Dublin Core. Além disso, foi definido o procedimento de uso desta linguagem em servidores Z39.50.

ABSTRACT

The number of organizations which create their own digital libraries has been growing since digital libraries offer an efficient method of organization in the storage of their stock as well as a facility in the finding of information. The value of all the knowledge stored in digital libraries is recognized when it becomes available for research. Thus, the interoperability between digital libraries is an important issue to address. Some initiatives have been proposed to deal with this problem, such as the Z39.50 protocol and the Dublin Core metadata standard. However, there are many implementations of proprietary digital libraries which do not adopt such procedures. Nowadays, there are solutions which allow proprietary libraries to adopt interoperability standards. But the installation of such solutions is complex, demanding even recodification. The aim of this study is to propose a solution to make it easier to adapt Z39.50 servers to proprietary digital libraries which use related databanks. To this end, a language based on XML was defined for the definition of the correlation between the metadata of proprietary digital libraries and the Dublin Core metadata. Furthermore, the procedure for the use of this language in Z39.50 servers was established.

INDICE

Capítulo 1. Introdução

Nos últimos tempos a Web vem tornando-se um meio eficaz de disseminar tecnologias e conhecimento, disponibilizando um imenso acervo cultural e científico para um público cada vez maior e mais heterogêneo. As informações digitalizadas estão espalhadas por milhares de servidores e torna-se premente a utilização de bibliotecas digitais, de forma a organizar o armazenamento da informação e facilitar sua recuperação.

Segundo a Digital Library Federation (DLF, 1998), bibliotecas digitais são organizações que provêm os recursos, inclusive o pessoal especializado, para selecionar, estruturar, oferecer acesso intelectual, interpretar, distribuir, preservar a integridade, e assegurar a constância com o passar do tempo, de coleções de trabalhos digitais de forma que eles estejam prontamente e economicamente disponíveis para uso por uma comunidade ou por um grupo de comunidades.

As bibliotecas digitais podem ser vistas como a evolução tecnológica das bibliotecas convencionais. O acervo que era armazenado nos mais variados tipos de mídias (papel, películas, fitas magnéticas, microfilme etc.) passa a sua forma digital. As bibliotecas digitais permitem: a preservação do acervo, a facilidade de busca e recuperação, e a disponibilização da informação para diversas comunidades espalhadas pelo mundo. As bibliotecas digitais estão geograficamente dispersas, e são integradas usando a tecnologia de transmissão de dados de que hoje dispomos.

A interoperabilidade entre bibliotecas digitais ainda não foi plenamente alcançada, em função de fatos impeditivos tais como: a falta da adoção de padrões de metadados e de protocolos de consulta e recuperação da informação. Como implicação percebemos uma sensível dificuldade em confirmar a proposta das bibliotecas digitais, qual seja: “transmitir informações para qualquer um, em qualquer lugar, a qualquer tempo” (POHLMANN, 1998).

Os padrões adotados pelas bibliotecas convencionais deveriam ter sido adotados por todos aqueles que publicam suas informações na Internet. Estes padrões permitem, por exemplo, organizar todo o acervo em fichários padronizados, que contêm

dados a respeito dos documentos que compõem o acervo. Estes dados que dizem respeito aos documentos são chamados de metadados (DCMI, 2003). Quando se faz necessária uma busca por assunto, autor ou tema (que seriam os metadados) basta pesquisar nestes metadados e assim obter uma lista mais concisa de documentos satisfazendo os critérios de busca, sendo que estes documentos são efetivamente relevantes para o usuário.

A iniciativa Dublin Core (DCMI, 2005) deu uma importante contribuição para a interoperabilidade entre as bibliotecas digitais. Ela criou um fórum aberto de discussão para estabelecer um padrão de metadados que fosse adotado por todos aqueles que publicam suas informações na Internet. Foi estabelecido o padrão de metadados Dublin Core, que pode ser hospedado em um documento XML e com isso tem-se garantido o intercâmbio de metadados.

O padrão Dublin Core é uma linguagem composta de 15 elementos (propriedades dos recursos) e qualificadores (adjetivos que contribuem para melhorar ou tornar mais preciso o significado das propriedades) que são usados para descrever, de forma padronizada, os mais variados recursos disponibilizados na Internet. Em outras palavras Dublin Core é uma linguagem fácil e inteligível, capaz de criar um catálogo digital dos recursos publicados na Internet e neste catálogo teríamos os recursos descritos, classificados e qualificados (Baker, 2000). Este padrão está sendo adotado por várias instituições de pesquisa, indústrias, universidades e comunidades científicas em todo o mundo, tornando-se assim um padrão verdadeiramente aceito.

O Z39.50 (ANSI, 1995) é um protocolo, reconhecido pela ANSI/NISO, para busca e recuperação de informações em base de dados. Ele opera através de uma arquitetura cliente/servidor e especifica procedimentos e formatos para a estação cliente pesquisar um banco de dados baseado num servidor. Possui as seguintes funcionalidades: estabelecimento de conexão, envio de consultas para o servidor, recuperação do resultado de consultas, classificação dos resultados, eliminação do resultado de consultas, pesquisa a palavras-chave, controle de acesso, controle de recursos e finalização da conexão. O Z39.50 pode ter, opcionalmente, a funcionalidade de atualizar a base de dados no servidor. Para prover a interoperabilidade entre o cliente

e o servidor, o protocolo Z39.50 pode ser usado em conjunto com o padrão de metadados Dublin e a linguagem SQL.

Existem outras iniciativas de criação de mecanismos de interoperabilidade de bibliotecas digitais aquelas advindas da Open Archives Initiative (OAI, 2005), que desenvolve e encoraja padrões de interoperabilidade que contribuem para facilitar a disseminação eficiente de recursos na Internet. Ela é atualmente uma organização e um esforço explicitamente em transição, e é direcionada para explorar e tornar possível este novo e amplo contexto de aplicações. Esta organização definiu o protocolo Open Archives Initiative Protocol for Metadata Haversting (OAI, 2004), referenciado por OAI-PMH, que por sua vez define um mecanismo para a coleta de metadados de Bibliotecas Digitais, apresentados no formato XML/Dublin Core.

O protocolo OAI-PMH permite realizar a colheita (*harvesting*) de metadados, ele não provê um mecanismo para a recuperação do conteúdo dos dados em si, este protocolo também não determina o significado de associações entre os metadados e conteúdos correlatos. Desde que muitos clientes podem necessitar acessar o conteúdo associado com os metadados colhidos, os provedores de dados podem julgar apropriado definir um link para o conteúdo do metadado. O padrão Dublin Core provê um identificador de elementos que podem ser usados para este propósito.

O padrão de metadados Dublin Core, a arquitetura OAI-PMH e o protocolo Z39.50 deram uma importante contribuição no sentido de prover a interoperabilidade entre as bibliotecas digitais. Contudo, existem diversas bibliotecas digitais, que não estão em conformidade com o padrão de metadados Dublin Core e o protocolo Z39.50, que são amplamente utilizados. Denominadas aqui de bibliotecas digitais proprietárias, as mesmas podem possuir um acervo amplo e significativo para comunidades acadêmicas e cientificas espalhadas pelo mundo. Ao conteúdo deste acervo, não deveria haver barreiras (tais como a falta de um padrão de metadados) que dificultem a recuperação da informação nele contida.

Algumas iniciativas visam integrar suporte ao protocolo Z39.50 às bibliotecas proprietárias, como os servidores Z39.50 JZKit (Ibbo, 2000), DBISERVER (Taylor, 2002), e YAZ (Hammer, 2000). Contudo, a atividade de adaptar qualquer um destes

sistemas para uma determinada biblioteca proprietária é complexa. Geralmente, esta atividade requer um trabalho de recodificação do código do servidor, a fim de adaptar o esquema do banco de dados proprietário no esquema compreendido pelo servidor Z39.50.

Poucas iniciativas existem para facilitar este trabalho de adaptação do servidor Z39.50 ao banco de dados proprietário. (Velegrakis, 2005) defende o uso de uma linguagem declarativa, chamada de *Description Language FrameWork*, para representação e conhecimento das classes de objetos e seus relacionamentos, que permite efetuar uma validação formal da eficiência com o que *wrapper(tradutor)* produzido é capaz de operar, ou seja, a definição de mapeamentos precisos e *Acess Points* (pontos de acesso aos dados) bem definidos. Contudo, esta linguagem não é de fácil utilização e requer um nível elevado de estudo antes da sua utilização prática. Sua complexidade pode ser equivalente a codificação direta do mapeamento.

1.1 Objetivo da Dissertação

O objetivo desta dissertação é propor uma solução para facilitar a adaptação de servidores Z39.50 a bibliotecas digitais proprietárias que utilizam banco de dados relacional. Para tal, foi definida uma linguagem baseada em XML para a definição da correlação entre metadados de bibliotecas digitais proprietárias em metadados Dublin Core. Esta linguagem, que chamaremos de DCM-BIB (Descritivo de Conversão de Metados para Bibliotecas digitais), é de fácil utilização e não requer que o administrador da biblioteca digital tenha habilidades de um programador para descrever a correlação entre os metadados.

Além da linguagem DCM-BIB, este trabalho define o procedimento de utilização das correlações definidas para a conversão da primitiva de busca Z39.50 para comandos SQL adequados ao banco de dados da biblioteca digital. Tal procedimento pode ser usado para a implementação de um novo servidor Z39.50, ou ainda ser utilizada por implementações do Z39.50 de código aberto que permitam a utilização do arquivo de correlação produzido pela linguagem DCM-BIB.

Para facilitar ainda mais a atividade de descrever a correlação entre os metadados Dublin Core e os metadados proprietários, foi definida uma interface gráfica que utiliza um formulário que é preenchido pelo Administrador da biblioteca digital e gera automaticamente um documento XML com a descrição DCM-BIB.

Para testar a linguagem DCM-BIB e a eficiência do algoritmo de mapeamento de metadados Dublin Core para os metadados proprietários, foi implementado um protótipo com uma interface gráfica que permitiu que fossem efetuados vários ensaios (formulação de consultas do Z39.50 e obtenção do correspondente em SQL, com base no mapeamento definido através da DCM-BIB) contra uma base de dados, que representa um suposto acervo de uma biblioteca digital proprietária.

1.2 Estrutura da Dissertação

O restante deste documento é organizado na forma que segue. O capítulo 2 apresenta alguns conceitos relacionados às Bibliotecas Digitais. O capítulo 3 apresenta a linguagem XML. Na seqüência, o capítulo 4 apresenta o padrão de representação de metadados Dublin Core. O capítulo 5 apresenta o protocolo Z39.50 e suas funcionalidades. O capítulo 6 apresenta a linguagem DCM-BIB e seu uso no servidor/Gateway Z39.50. E finalmente, o capítulo 7 apresenta as conclusões deste trabalho.

Capítulo 2. Bibliotecas Digitais

Historicamente, os sistemas de distribuição de informação, por meio de redes de computadores, foram desenvolvidos por comunidades científicas que tinham por objetivo o intercâmbio de informações entre seus membros. As bibliotecas digitais, utilizando a grande rede de computadores que é a Internet, estão inclusas num contexto socioeconômico e legal, bem mais amplo e complexo, que impõe à sua arquitetura um princípio de respeito aos direitos autorais. Os artistas, músicos, escritores, editoras, produtoras e gravadoras de forma geral, se mantêm pelo que arrecadam através do recebimento sobre cópia e distribuição do material produzido. Esta produção cultural estará inclusa nas BD, se as mesmas obedecerem ao princípio dos direitos autorais.

Este capítulo apresenta alguns conceitos relacionados às bibliotecas digitais.

2.1 Definição de Biblioteca Digital

A seguir serão apresentadas alguns definições e requisitos apresentados por diversos autores.

A *Digital Library Federation* (DLF, 1998) define bibliotecas digitais como organizações que provêm os recursos, inclusive o pessoal especializado, para selecionar, estruturar, oferecer acesso intelectual, interpretar, distribuir, preservar a integridade, e assegurar a constância com o passar do tempo, de coleções de trabalhos digitais de forma que eles estejam prontamente e economicamente disponível para uso por uma comunidade ou por um grupo de comunidades".

A *Digital Libraries Initiative* (DLI, 1998) afirma que uma biblioteca digital não é somente o equivalente a conjuntos digitalizados com métodos de gestão da informação, mas também um conjunto de coleções, serviços e pessoas que favorecem o

ciclo completo da criação, difusão, uso e preservação dos dados, para a informação e para o conhecimento.

Pullian (1996) define brevemente biblioteca digital como sendo uma infra-estrutura de informações eletrônicas, em forma padronizada, que permite: a) o armazenamento distribuído de dados sobre uma região geograficamente grande, b) procura e acesso através de elos (*links hipertextuais*) e c) operações transparentes ao usuário final. Em termos mais simples, bibliotecas digitais poderiam ser definidas como sistemas capazes de armazenar dados em vários *sites* e fornecer ao usuário uma interface para a procura de informações sobre estes vários repositórios em um único passo.

Idealmente, uma biblioteca digital deveria transmitir informações para qualquer um, em qualquer lugar, a qualquer tempo. Mas muitas redes de computadores de hoje não satisfazem o requisito de grande largura de banda das bibliotecas digitais (Pohlmann, 1998).

A maioria das definições de bibliotecas digitais identificam que (SunSite, 1995):

- Biblioteca digital não é uma entidade única.
- Biblioteca digital requer tecnologia para ligar recursos distribuídos.
- As ligações entre as várias bibliotecas digitais e serviços de informação são transparentes ao usuário final.
- Acesso universal às bibliotecas digitais e serviços de informação é uma meta.
- Coleções das bibliotecas digitais não são limitadas a armazenamento de documentos textuais, gráficos e imagens; elas incluem outros tipos de meios de apresentação que não podem ser representadas ou distribuídos no formato impresso, como áudio e vídeo.

2.2 Atividades para criação e manutenção de uma Biblioteca Digital

As atividades envolvidas na criação e manutenção de uma biblioteca digital são (Pullian, 1995): criação e captura, gerenciamento e armazenamento, busca e acesso, disponibilização e tratamento de direitos autorais.

2.2.1 Criação e Captura

Esta atividade consiste na análise e seleção dos objetos a serem disponibilizados na biblioteca digital. Os objetos que não estão digitalizados passam por um processo de conversão para o formato digital, a isto denominamos captura/transformação. Por exemplo, uma foto antiga, um filme em película, um manuscrito etc.

2.2.2 Gerência e armazenamento

O acervo multimídia normalmente requer grandes espaços de armazenamento, a previsão de um crescimento sensível ao longo do tempo, e a necessidade de preservação indefinida. Diante destes requisitos, torna-se necessário a definição de dispositivos de armazenamento de alta performance, e de preferência distribuídos entre servidores distintos. Além disto, deve-se definir procedimentos de backup automático e redundantes.

2.2.3 Busca e Acesso

As técnicas de indexação convencionais são utilizadas para facilitar a busca e recuperação do acervo. Nos índices temos os metadados, que caracterizam os objetos do acervo, viabilizando assim as buscas convencionais pelos mais variados atributos (autor, título, assunto, palavras-chaves, cor, forma, textura etc). Basicamente duas arquiteturas podem ser usadas para a construção dos índices. A primeira utiliza um banco de dados relacional e nele são mantidos os índices de metadados, que por sua vez apontam para um diretório de um sistema de arquivo convencional, onde os objetos (textos, imagens, sons) residem. A segunda também utiliza um índice de metadados, com a diferença de que os objetos são armazenados dentro do banco de dados.

2.2.4 Distribuição

Para que seja possível a disponibilização ampla, irrestrita e contínua do acervo de uma biblioteca digital, se faz necessário o planejamento minucioso da infra-estrutura física que irá suportar a mesma. Desta maneira, é fundamental que haja servidores de alta disponibilidade, fornecimento ininterrupto de energia elétrica, temperatura e umidade controladas, além de uma rede de transmissão de dados com baixo índice de inoperância.

2.2.5 Tratamento de direitos autorais

Trata-se dos mecanismos que irão reforçar o respeito aos direitos autorais e regras de distribuição das obras digitalizadas. Este assunto ainda não está claramente definido no mundo Web. Como exemplo, poderíamos citar as músicas digitalizadas em formato MP3.

2.3 Interoperabilidade em Bibliotecas Digitais

A falta de interoperabilidade entre softwares e repositórios de domínios diferentes é a maior barreira para o intercâmbio do conteúdo digital. Nas bibliotecas digitais, os níveis de interoperabilidade podem incluir a federação, coleta e Busca (Lightle, 2003).

A federação (Gonçalves, 2001) provê a forma mais forte de interoperabilidade, mas exige um grande esforço a seus participantes.

A Busca não impõe aos participantes nenhum esforço, mas provê um pobre nível de interoperabilidade.

O conceito de coleta é que os participantes concordam em efetuar um pequeno esforço que habilita alguns serviços básicos, sem requerer a adoção de um conjunto de conformidades.

Alguns esforços existem para promover e desenvolver padrões de interoperabilidade que facilitam a coleta de metadados. A solução proposta permite o

intercâmbio de catálogos e até mesmo uma linguagem padrão para a recuperação da informação.

A iniciativa Dublin Core (DCMI, 2005) é uma organização dedicada a promover a disseminação de padrões de metadados interoperáveis e o desenvolvimento de vocabulários de metadados especializados na descrição de recursos que tornam os sistemas de recuperação da informação mais inteligentes. Os metadados são dados sobre os dados. O Dublin Core permite especificar informações que expressam o conteúdo intelectual, a propriedade intelectual, e/ou a exposição das características de uma coleção digital.

O Open Archives Initiative (OAI) (OAI, 2005) define o protocolo OAI para a coleta de metadados, referenciado como PMH-OAI, que prove um suporte a interoperabilidade baseado na coleta de metadados de repositórios baseados em documentos XML. As bibliotecas digitais que adotam o PMH-OAI podem atuar como provedores de dados e seus metadados podem ser colhidos através de um serviço. Este serviço pode ser um portal de acesso para todas as bilbiotecas provedoras criando assim uma confederação de bibliotecas digitais. O projeto BDB (Biblioteca digital Brasileira) (BDB, 2005) adota o OAI para integrar em um único portal o mais importante repositório digital do Brasil, de forma a permitir uma recuperação de informação unificada e simultânea a este repositório.

2.4 Conclusão

Este capítulo apresentou uma definição de biblioteca digital, seu propósito, classificação e atividades de criação e manutenção da mesma. Podemos também concluir, que a biblioteca digital é um importante meio de democratizar o acervo cultural, tecnológico e científico produzidos por uma comunidade. O próximo capítulo apresenta a linguagem XML.

Capítulo 3. A LINGUAGEM XML

Este capítulo tem por objetivo apresentar a linguagem XML, mostrando principalmente os conceitos básicos necessários para o entendimento da linguagem DCM-BIB proposta nesta dissertação. Esta linguagem é usada para descrever a correlação entre os metadados Dublin Core e proprietários e a relação entre as tabelas de um banco de dados que armazena um acervo digital de uma biblioteca.

3.1 Introdução

A Web cresceu rapidamente e estendeu seu propósito inicial (publicação de textos científicos) para se tornar um meio de comunicação completo e abrangente além de ser interativa. Para conseguir atender a todas as aplicações, o número de tags da HTML, cresceu de uma dúzia (na versão inicial) para mais de cem tags (na versão 4.0). Apesar do crescimento do número de tags, a demanda por mais tags continua, de acordo com a variabilidade das aplicações atualmente em uso.

O HTML chegou em um ponto insustentável, ou seja, se continuar crescendo pode acabar por prejudicar alguns tipos de aplicação. Para atender a esta demanda e resolver os problemas da HTML, surge a proposta XML (*eXtensible Markup Language*). XML (XML, 2005a) é uma linguagem de marcação de texto desenvolvida pelo consórcio W3C. Esta linguagem é uma proposta para solucionar as limitações da tão disseminada HTML.

Em XML, não se predefine tags e usa-se uma sintaxe mais restritiva. Com esta proposta é possível definir as tags que forem necessárias para o documento a ser publicado. Por isso é dito que XML é extensível. Uma sintaxe mais restritiva torna os navegadores mais simples, menores e mais velozes. Ao contrário da HTML que tem uma sintaxe mais permissiva e obriga os navegadores a tratar erros de quem define o

documento a ser publicado. A XML basicamente atende a duas classes de aplicações: Aplicações de documentos, voltadas para o ser humano; e aplicações de dados, direcionadas para o consumo do software. Embora sejam distintas do ponto de vista da sua finalidade, estas classes usam a mesma linguagem XML, ou seja, consegue-se manter um padrão e isto se traduz em um aproveitamento e reusabilidade do documento XML.

As aplicações de documentos são o alvo principal da linguagem XML, que foca sua atenção na estrutura do documento. Esta característica cria uma independência do meio de distribuição, onde o documento será apresentado. Assim é possível publicar documentos XML automaticamente em dispositivos variados, usando as linguagens HTML, PostScript, XHTML, PDF etc.

Esta aplicação de dados com XML torna-se possível uma vez que conseguimos representar a estrutura de um banco de dados, ou seja, suas tabelas, colunas e linhas em um documento estruturado XML. Outras aplicações podem ler este documento e com isto é possível integrar as mais diversas aplicações. Tomamos como exemplo um fornecedor de produtos farmacêuticos, que possui milhares de itens em estoque. Este fornecedor pode publicar, usando XML, a relação de produtos, preços, prazo de entrega e quantidade disponível na Internet. Os aplicativos que executam no cliente, no caso nas farmácias, podem acessar automaticamente este documento XML, via Internet, e atualizar suas tabelas locais, filtrando aquilo que é de seu interesse.

3.2 Sintaxe da XML

A linguagem XML concentra sua atenção na estrutura do documento. Um documento XML é composto de dados caracteres e marcação. O importante para o ser humano que lê o documento são os dados representados por caracteres. Já a importância da marcação é percebida quando torna-se necessário o processamento automático do documento. Neste caso, o software que processa o documento precisa saber onde começa e termina o dado, e também qual o seu significado.

3.2.1 Tags de Início e Fim

Um **elemento** é a unidade básica de um documento XML. Um elemento é composto de um **tag de início**, um texto e um **tag de fim.** O tag de inicio identifica o nome do tag, entre os sinais "<" e ">". Um tag de fim, inclui um caractere "\" antes do nome. Por exemplo, em *<nome>João da Silva<\nome>*, *<nome>* é a tag de início, *João da Silva* é o dado, e *<\nome>* é a tag de fim.

3.2.2 Nomes

Os nomes em XML têm que começar com uma letra ou caractere de sublinhado "_". O restante do nome pode ser constituído de letras, dígitos, etc. Os nomes não podem começar com XML.

3.2.3 Atributos

Os atributos associam informações aos elementos. Os elementos podem ter um ou mais atributos no tag de início. O atributo é separado da informação pelo sinal de "=". Por exemplo:

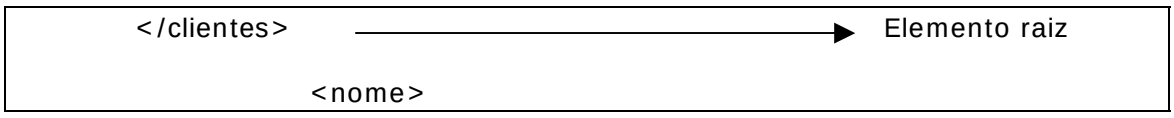
```
<comida tipica= "Chinesa">frango xadrez</comida>
<comida tipica= "Brasileira">arroz e feijão</comida>
```

A informação do atributo deve vir sempre entre aspas ou apóstrofe.

3.2.4 Árvore de elementos

Os elementos em um documento XML, podem se organizar como uma árvore. Não há limites para a profundidade da árvore. Os elementos também podem se repetir.

Um elemento interno a outro elemento é chamado de filho, o elemento externo a ele é chamado de pai. Os tags de inicio e fim dos elementos filhos, devem estar no escopo dos tags do elemento pai. Além disso, o elemento raiz deve ser único no documento. Por exemplo:



```
                        <primeironome>Fernando</ primeironome>
                        <ultimonome>Maia</ultimonome>
            </nome>
                    <nome>
                        <primeironome>André</ primeironome>
                        <ultimonome>Lucena</ultimonome>
            </nome>
        </clientes>
```

3.2.5 Declaração básica de um documento XML

A primeira linha de um documento XML deve conter :

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

Esta declaração indica que trata-se de um documento XML e sua versão. Veja o documento abaixo :

```
<?xml version="1.0" enconding="ISO-8859-1"?>
<veiculosemestoque>
   <unidade>
      <marca>GM</marca>
      <modelo>Astra </modelo>
      <cor>Prata</cor>
      <ano>2000</ano>
   </unidade>
   <unidade>
      <marca>GM</marca>
      <modelo>Vectra </modelo>
      <cor>Azul</cor>
      <ano>1999</ano>
   </unidade>
   <unidade>
      <marca>VW</marca>
      <modelo>Gol </modelo>
      <cor>Brancol</cor>
      <ano>2003</ano>
      <direcao>hidraulica</direcao>
      <ar>refrigerado</ar>
   </unidade>
</veiculosemestoque>
```

3.2.6 Comentários

Os comentários começam com "!- -" e terminam com "- ->". Devem vir antes ou após uma marcação. Exemplo :

```
        <! - - descrição sucinta dos produtos de acordo a norma x -->
```

3.3 XML e Padrões Acompanhantes

A Linguagem XML não é apenas uma linguagem de marcação, ela é de fato uma linguagem de marcação padrão. O W3C normatiza a linguagem e desenvolveu alguns padrões que complementam a XML, os denominados padrões acompanhantes.

O Padrão XML Namespace resolve um problema que surge naturalmente por efeito da extensibilidade da linguagem, como os tags podem ser livremente criados, podem surgir conflitos que tornam o documento XML confuso e impreciso. O padrão Namespace tem como finalidade evitar a duplicidade dos tags, uma vez que associa uma URI exclusiva ao tag. Uma namespace é definida da seguinte forma : xmlns : xx = "URI" onde xmlns é um termo reservado do XML, xx é um prefixo associado a URI.

Por exemplo:

```
<?xml version= "1.0" enconding= "ISO-8859-1"? >
<referências    xmlns:ns= "http://solution.all.com/definicao/doc.1"
                xmlns:ps= "http://outrasolution.all.com/definicao/doc.21"
                xmlns= "http://padraogeral.com/default/pdoc1">
<nome>Empresa Um</nome>
<ns:status>Operante</ns:status>
<ps:status>Inadimplente</ns:status>
<nome>Empresa Dois</nome>
<ns:status>Operante</ns:status>
<ps:status>adimplente</ns:status>
<nome>Empresa Três</nome>
<ns:status>Operante</ns:status>
<ps:status>Pendentee</ns:status>
</referencias>
```

Observamos que sem os prefixos "ns" e "ps", predefinidos e associados a um domínio, teríamos a ambigüidade dos tags "status", como seria possível interpretar a informação "Operante" ou "Inadimplente"? A empresa teria dois status ? . O parser XML substitui os prefixos "ns" e "ps" pelas respectivas URIs que garantem a exclusividade dos nomes.

O Padrão Folhas de Estilo determinam como os documentos XML devem ser apresentados no browser, no papel ou em editores. A XML trabalha com duas linguagens de estilo: XSL (XML Stylesheet Language) (W3C, 2005b) e CSS (Cascading Style Sheet) (W3C, 2005c).

DOM (Document Object Model) (W3C, 2005d) e SAX (Simple API for XML) (SAX, 2005) tratam-se de duas APIs que permitem que o documento XML seja acessado por outros aplicativos que não reconhecem a sintaxe XML.

Xlink (W3C, 2005f) e XPointer (W3C, 2005g) são partes de um padrão, que permite a associação de documentos, ou seja, permitem o estabelecimento de ligações de um documento XML com outros recursos disponíveis na Web.

3.4 O esquema XML

O esquema XML (W3C, 2001b) é uma alternativa a DTD (*Document Type Definition)* e é referenciados por XSD (XML Schema Definition). O propósito do XSD é definir a estrutura de um documento XML como faz a DTD. Na realidade o XSD é um sucessor da DTD e tornou-se uma recomendação da W3C em maio de 2001. A Figura 1 mostra um equema XML. Como pode ser visto, o elemento <schema> é o elemento raiz. O fragmento "xmlns:xs=http://www.w3.org/2001/XMLSchema" indica que os elemento e o tipo vem de http://www.w3.org/2001/XMLSchema. O fragmento http://www.w3schools.com indica que os elementos (note, to, from, heading, body) deste namespace. O fragmento elementFormDefault="qualified" indica que qualquer documento XML instanciado a partir deste esquema deve ser qualificado pelas namespaces declaradas no esquema.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
<xs:element name="note">
   <xs:complexType>
     <xs:sequence>
   <xs:element name="to" type="xs:string"/>
   <xs:element name="from" type="xs:string"/>
   <xs:element name="heading" type="xs:string"/>
   <xs:element name="body" type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
</xs:element>
</xs:schema>
```

Figura 1. Um esquema XML

A linguagem XML, permite o intercâmbio de dados entre as mais variadas aplicações. Utilizando o esquema XML, pode-se definir previamente a estrutura de um documento e garantir que o intercâmbio de dados possa ser preciso e compreensível entre quem envia e quem recebe o documento XML.

3.5 Conclusão

Neste capítulo foi feita uma breve apresentação da linguagem XML, sua estrutura, sintaxe e padrões acompanhantes. O próximo capítulo apresenta o padrão de metadados Dublin Core.

Capítulo 4. O PADRÃO DUBLIN CORE

A iniciativa Dublin Core (DCMI - *Dublin Core Metadata Initiative*) (DCMI 2005) pode ser vista como um fórum aberto de discussão que tem como objetivo estabelecer um padrão de metadados, usando XML, amplamente aceito, para suportar e facilitar a busca de recursos na Internet. Outro importante aspecto, considerado por este fórum, é a facilidade em prover a interoperabilidade.

O escopo de atividades da DCMI inclui: o desenvolvimento de um padrão de metadados para a pesquisa e recuperação de dados entre as mais variadas áreas do conhecimento humano; a definição de mecanismos de interoperabilidade dos metadados; o desenvolvimento de metadados especializados para comunidades científicas entre outras, tal como o DCMI Working Group Libraries que desenvolve um trabalho específico na área de bibliotecas digitais.

Este capítulo descreve o padrão Dublin Core, mostrando seu histórico, componentes e seu propósito.

4.1 Definição de Metadados

Metadados são dados a respeito de dados. No contexto do Dublin Core, é a informação que qualifica a descrição de um recurso na Internet. Funcionalmente, seria uma estrutura de dados composta de elementos necessários para descrever de forma clara e objetiva um recurso. Por analogia, em uma biblioteca convencional temos um catálogo composto de um conjunto de fichas. Estas fichas contêm dados a respeito de um livro, a saber: Autor, assunto, editora, data de criação, posição na prateleira, etc. estes dados contidos nas fichas são os metadados do documento. Desta forma, um metadado na Internet, poderia ser imaginado como um catálogo eletrônico onde teríamos um “fichário” que descreve recursos na Internet.

O grande volume de publicações eletrônicas na Internet tem despertado o interesse mundial em padrões de metadados, que tem como principal objetivo aumentar a capacidade de localização e recuperação de recursos relevantes para a comunidade de usuários da Internet.

“A associação de padrões de descrição de metadados com objetos de rede tem potencial e substancialmente aumentando a capacidade de descoberta de recursos pela habilitação de pesquisas de campo (autor, título), permitindo a indexação de objetos não-textuais, e permitindo o acesso de conteúdo representativo do recurso que é diferente de acessar o conteúdo de recurso em si” (Weibel, 1997).

4.2 Dublin Core

O padrão de metadados Dublin Core (DC) é uma linguagem. Esta linguagem é composta de elementos, mais precisamente 15 elementos (nomes), e qualificadores (adjetivos) que são usados para descrever os mais variados recursos da Internet. A semântica destes elementos tem sido estabelecida consensualmente por várias comunidades internacionais. Outra definição seria uma “pequena linguagem para elaboração de uma classe particular de declarações sobre recursos” (Baker, 2000). Por este ponto de vista, esta “pequena linguagem” traduz a idéia de simplicidade e facilidade de uso para usuários da Internet, os quais pretendem encontrar o que procuram em uma diversidade de recursos que normalmente estão em variados idiomas.

“O Dublin Core Metadata Element Set (DCMES), pode ser visto como blocos de metadados Web. Seus 15 elementos são úteis para a criação simples, e de fácil entendimento, da descrição dos recursos da Web” (Weibel, 2000). Estes blocos de metadados possibilitam a catalogação dos recursos visando principalmente facilitar a busca e recuperação dos mesmos. Outro aspecto que deve ser notado, é que os blocos de metadados podem ser combinados para formar descrições complexas que por sua vez podem atender os requisitos funcionais de aplicações diferentes.

Estes blocos de metadados possibilitam a catalogação dos recursos visando principalmente facilitar a busca e recuperação dos mesmos. Outro aspecto que deve ser

notado, é que os blocos de metadados podem ser combinados para formar descrições complexas que por sua vez podem atender os requisitos funcionais de aplicações diferentes.

4.3 A gramática Dublin Core

Na gramática Dublin Core, os termos são definidos e classificados, depois a estrutura da frase é examinada. No Dublin Core, elementos e qualificadores são agrupados para a construção do equivalente de metadados a frases e parágrafos.

4.3.1 Vocabulário de Termos

Um elemento ou qualificador Dublin Core é um identificador único composto por um nome prefixado por uma URI de *namespaces*. Por exemplo, htttp://dublincore.org/2000/03/13-dces#title. Neste contexto, um namespace é um vocabulário que aponta para um recurso na Web. Ele descreve o recurso em linguagem natural e com conteúdo relevante para a pesquisa e recuperação.

4.3.2 Elementos

Os elementos DC seriam as propriedades do recurso. Com eles podemos criar catálogos úteis para a pesquisa em vários domínios do conhecimento. Os quinze elementos, ou nomes, que o Dublin Core define são:

- **dc:title:** indica o título do recurso, normalmente dado pelo editor ou autor.
- **dc:creator**: indica a pessoa ou organização que possui a propriedade intelectual do recurso.
- **dc:subject**: descreve o conteúdo do recurso, ou seja, vocabulários, palavras chaves ou frases que descrevem o recurso.
- **dc:source:** trata-se de uma referência a fonte do qual o recurso possa ter sido derivado.

- **dc:description**: a descrição textual do recurso.
- **dc:publisher**: a entidade que mantém o recurso, tal como uma universidade, corporação ou Editora.
- **dc:contributors**: pessoa ou organização que contribuíram para a elaboração do recurso juntamente com o dc:creator.
- **dc:date**: a data que o recurso foi criado ou disponibilizado na sua forma atual.
- **dc:type**: o tipo do recurso, como por exemplo, uma página em html, um trabalho técnico, um ensaio, um dicionário etc. Os tipos disponíveis são : “collection”, “dataset”, “event”, “image”, “interactive resource”, “service”,”software”, “sound”, “text”, “physicalObject”. (DCMI 2002).
- **dc:format**: o formato de representação do recurso, tal como um html, um txt, uma imagem em jpeg.
- **dc:identifier:** identifica de forma única o recurso. Como por exemplo, um URL.
- **dc:language**: identifica a linguagem na qual o recurso foi produzido.
- **dc:relation**: indica o relacionamento com outros recursos de conteúdos similares ou complementares ao assunto.
- **dc:coverage**: referencia uma localização geográfica, um período de tempo ou uma jurisdição.
- **dc:rights**: sua função é de ser um link para um servidor que dê informações de direitos autorais.

Na Figura 2 é apresentado um exemplo dos uso dos elementos DC em um documento XML.

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

```
 xmlns="http://example.org/myapp/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://example.org/myapp/
http://example.org/myapp/schema.xsd"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:dcterms="http://purl.org/dc/terms/">

 <dc:title>
  UKOLN
 </dc:title>
 <dcterms:alternative>
  UK Office for Library and Information Networking
 </dcterms:alternative>
 <dc:subject>
  national centre, network information support, library
  community, awareness, research, information services,public
  library networking, bibliographic management, distributed
  library systems, metadata, resource discovery,
  conferences,lectures, workshops
 </dc:subject>
 <dc:subject xsi:type="dcterms:DDC">
  062
 </dc:subject>
 <dc:subject xsi:type="dcterms:UDC">
  061(410)
 </dc:subject>
 <dc:description>
  UKOLN is a national focus of expertise in digital information
  management. It provides policy, research and awareness services
  to the UK library, information and cultural heritage communities.
  UKOLN is based at the University of Bath.
 </dc:description>
 <dc:description xml:lang="fr">
  UKOLN est un centre national d'expertise dans la gestion de l'information
  digitale.
 </dc:description>
 <dc:publisher>
  UKOLN, University of Bath
 </dc:publisher>
 <dcterms:isPartOf xsi:type="dcterms:URI">
  http://www.bath.ac.uk/
 </dcterms:isPartOf>
 <dc:identifier xsi:type="dcterms:URI">
  http://www.ukoln.ac.uk/
 </dc:identifier>
 <dcterms:modified xsi:type="dcterms:W3CDTF">
  2001-07-18
 </dcterms:modified>
 <dc:format xsi:type="dcterms:IMT">
  text/html
 </dc:format>
 <dcterms:extent>
  14 Kbytes
 </dcterms:extent>

</metadata>
```

Figura 2. Exemplo do uso dos elementos Dublin Core em um documento XML

4.3.3 Qualificadores

Como os adjetivos da linguagem natural, os qualificadores modificam os elementos tornando seu significado mais claro. Os qualificadores estão em duas classes: **Esquemas Codificados** e **Refinamento de Elementos**.

Os **Esquemas Codificados** são indicadores de informações contextuais (dcq-Dublin core qualifier) que auxiliam a compreender o valor de um elemento. Exemplo dcq:SUS qualifica dc:subject “Atendimento Médico de Qualidade” por indicar a sigla SUS (Sistema Único de Saúde), ou seja, contextualiza dc:subject. Outro Exemplo : dc:date “2000-06-13” e dcq: iso8601 qualifica dc:date por especificar que a data é formada por ano, mês e dia, seguindo um padrão internacional.

O Refinamento de Elementos tornam a propriedade mais específica, por exemplo, dc:date e dcq:revised que indica que dc:date representa uma “data revisada” do recurso.

4.3.4 Outros Idiomas

A DCMI discute e define os elementos e qualificadores no idioma Inglês. Contudo, eles podem ser definidos sem problemas em outros idiomas. Exemplo: dc:creator poderia ser escrito dc:creatore em italiano ou dc:verfasser em Alemão. Para as datas os elementos já foram traduzidos em 26 línguas.

4.3.5 Declarações Dublin Core

Os diagramas usados para ensinar a gramática inglesa, que separa o sujeito do predicado, são usados como modelo para indicar como as declarações do Dublin Core são feitas. Neste diagrama o predicado, que diz algo a respeito do sujeito (em dublin core é o recurso), é separado deste por uma linha. No predicado, o objeto (em dublin core é a propriedade) é separado do complemento do objeto (no dublin core é o valor da propriedade) por uma barra

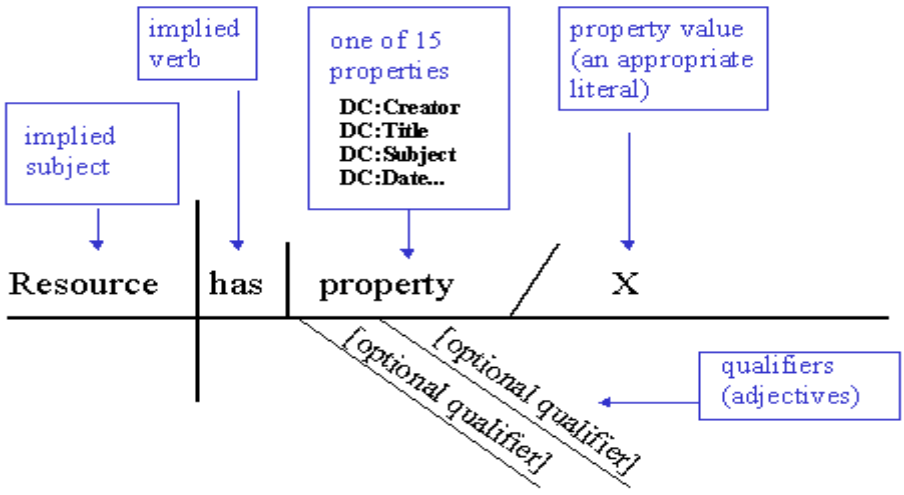


Figura 3. Diagrama de descrição de recurso (Baker, 2000)

Na figura 3 é mostrado um diagrama que indica como o Dublin Core descreve um recurso. Assim um "Recurso tem a propriedade X", onde a propriedade é um dos quinze elementos, e X é o valor da propriedade. Por exemplo : "Recurso tem dc:creator" "Carlos Drumond de Andrade", e "Recurso tem dc:date" "2000-06-13". Eventualmente os qualificadores podem fazer o significado de uma propriedade mais específico. Por exemplo: "Recurso tem dc:date" "2000-06-13" e "Recurso tem dcq:revised" "2001-06-23".

4.3.6 Princípio Dumb-down

Trata-se de uma regra para a interoperabilidade de qualificadores, onde os qualificadores podem tornar o significado do elemento mais preciso, contudo, não podem fazer parte do seu significado. Desta forma, se um qualificador for ignorado, isto só vai causar a perda de precisão, mas o resultado, ainda terá um significado útil para aplicação ou usuário.

4.3.7 Literais apropriados

O valor de uma propriedade ou elemento deve ser útil para facilitar a busca e recuperação do recurso que ela ajuda a descrever. Assim o valor dos elementos dc:creator, dc:contributor, dc:publisher, ou dc:title deveriam ser uma string; o valor dos

elementos dc:relation, dc:identifier, ou dc:source uma URL; o valor para dc:subject palavras chaves; o valor para dc:date uma data baseada em padrões internacionais.

4.4 Conclusão

Neste capítulo, foi apresentado o padrão de metadados Dublin Core. Visto sob o ponto de vista de uma linguagem, que possui nomes e adjetivos para descrever os mais variados recursos na WEB. As bibliotecas digitais, atualmente em uso, não adotam o Dublin Core como padrão para expor seus metadados a comunidade de usuários. Tal fato dificulta a busca, recuperação e o intercâmbio de informações entre as bibliotecas digitais e seus usuários. Esta dissertação pretende apresentar uma possível solução para esta deficiência. O próximo capítulo apresenta o padrão Z39.50.

Capítulo 5. O Padrão Z39.50

O padrão Z39.50 (ANSI, 1995) utiliza a abordagem cliente/servidor e fornece uma solução (constituída de serviços e protocolo) para a recuperação de informações. Ele opera na camada de aplicação, segundo o modelo de referência OSI e especifica procedimentos e formatos para a estação cliente pesquisar em um ou mais banco de dados baseado em um ou múltiplos servidores. Este padrão é amplamente utilizado no escopo das bibliotecas digitais.

O objetivo deste capítulo é o de apresentar as características e funcionalidades deste padrão como parte do embasamento teórico necessário para o entendimento da proposta desta dissertação.

5.1 Modelo de Funcionamento do Z39.50

Z39.50 define um modo padrão para dois computadores se comunicarem com a finalidade de recuperação de informação. A Figura 4 ilustra essa comunicação. Através da padronização de procedimentos e características, Z39.50 torna fácil a tarefa de pesquisar e recuperar informações em grandes bancos de dados. Especificamente, Z39.50 suporta recuperação de informação dentro um ambiente cliente/servidor distribuído, onde um computador que opera como um cliente submete um pedido de busca (uma consulta) para outro computador que age como um servidor de informação. Um software no servidor executa a busca em um ou mais bancos de dados e cria um conjunto de resultados composto de registros que atendem aos critérios do pedido de busca. O servidor retorna para o cliente o conjunto de resultados contendo registros para que o mesmo possa processá-los.

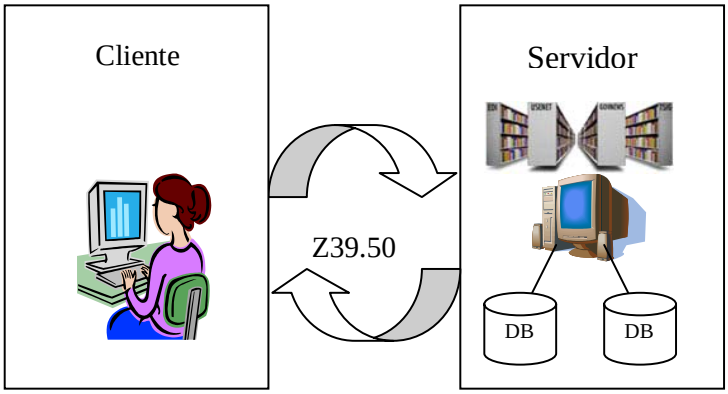


Figura 4. Comunicação entre Cliente e Servidor através do Protocolo Z39.50

O poder de Z39.50 é que ele separa a interface de usuário no lado cliente dos servidores de informação, mecanismos de busca e bancos de dados. Z39.50 provê uma visão consistente da informação a partir de uma ampla variedade de fontes, e oferece para os implementadores de cliente a capacidade de integrar informações a partir de um grande número de bancos de dados.

A comunicação entre cliente e servidor acontece através de uma *Z-Association*. Uma *Z-Association* é explicitamente estabelecida pelo cliente e pode ser terminada explicitamente pelo cliente ou servidor, ou implicitamente terminada através da perda de conexão. Os papéis de cliente (*Z-Client*) e servidor (*Z-Server*) dentro de uma *Z-Association* não podem ser invertidos. Uma Z-association é estabelecida pelo cliente e encerrada pelo próprio cliente ou pelo servidor. Em uma conexão entre o cliente e o servidor, pode-se estabelecer várias Z-associations. Em cada Z-association podem ocorrer operações concorrentes. Uma *Z-Association* não pode ser reiniciada, assim uma vez que uma *Z-Association* é terminada nenhuma informação de estado é retida, exceto a informação que é explicitamente salva.

O itinerário de um processo de busca (Bordin, 2002) envolvendo este protocolo se dá da seguinte forma (Figura 5):

- Usuário seleciona a biblioteca destino (*Z-Server*) a partir de um menu de uma aplicação;

- Usuário entra com os termos de pesquisa.

- Os termos de pesquisa e a biblioteca destino são enviados para o *Z-Client*
- *Z-Client* traduz os termos pesquisados para dentro de "*Z-Speak*" e contata o software *Z-Server* da biblioteca destino.
- Há uma negociação preliminar entre *Z-Client* e *Z-Server* para estabelecer as regras para a "*Z-Association*" entre os dois sistemas.
- *Z-Server* traduz o "*Z-speak*" para dentro de uma requisição de busca para o banco de dados da biblioteca destino e recebe a resposta sobre o número de comparações encontradas.
- *Z-Client* recebe os registros.
- Registros são apresentados para o usuário.

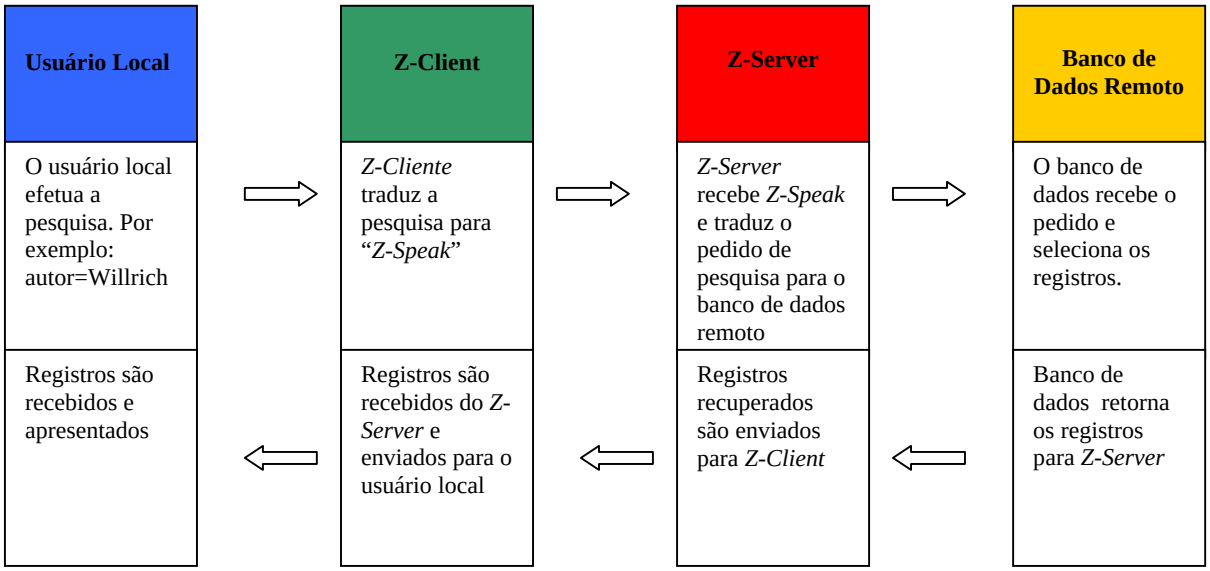


Figura 5. Itinerário de Busca de Informação através da Utilização do Protocolo Z39.50

5.2 Modelo da Base de Dados

O padrão Z39.50 tem por objetivo permitir que um cliente recupere as informações em um servidor. Existem diversas implementações possíveis para uma base de dados, tal fato sugere um ambiente computacional heterogêneo. Diante disto, o

padrão Z39.50 trabalha com um modelo abstrato que descreve um banco de dados genérico. Os diversos sistemas podem mapear suas características de implementação para este modelo. E com isto o padrão provê a interoperabilidade entre eles.

O padrão Z39.50 entende que uma base de dados é uma coleção de registros. Cada registro é o agrupamento de informações correlatas. Um registro da base de dados representa a estrutura de dados, que por sua vez, representa a informação. Os pontos de acesso seriam os índices que permitem o acesso direto aos registros da base de dados a partir dos elementos que constituem os registros (no modelo relacional seriam as colunas de uma tabela). Por exemplo, um ponto de acesso “autor” pode ser usado para recuperar registros de um autor específico, este ponto de acesso pode ser usado para a pesquisa em diversas bases de dados com implementações distintas.

5.3 Serviços Z39.50

Os serviços são executados através do intercâmbio de mensagens entre o cliente e o servidor. A mensagem, neste contexto, é uma solicitação ou a resposta de uma solicitação. São onze os serviços disponibilizados pelo padrão Z39.50, os mesmos são apresentados no quadro 1.

Os serviços Z39.50 possuem um status confirmado, não confirmado ou condicionalmente confirmado. Os serviços confirmados são aqueles onde há uma solicitação (do cliente ou do servidor) seguido de uma resposta correspondente. Um exemplo de um serviço confirmado seria o *search*. Os serviços não confirmados são aqueles solicitados pelo cliente ou servidor, sem resposta correspondente. Um exemplo de serviço não confirmado seria o *segment*. Um serviço condicionalmente confirmado é aquele que pode ser solicitado como um serviço confirmado ou não confirmado. Trata-se de uma solicitação (do cliente ou do servidor) que pode ou não ter uma resposta. Um exemplo deste tipo de serviço seria o Resource-Control.

Quadro 1. Serviços Z39.50

Serviço	Descrição
Initialization	**Init :** Um serviço confirmado invocado pelo cliente para inicio de uma operação.
Search	**Search:** Um serviço confirmado invocado pelo cliente para uma operação de pesquisa.
Retrieval	Possui dois serviços : • **Present :** Um serviço confirmado invocado pelo cliente para iniciar uma operação Present. Trata-se de um pedido de registros ao servidor. • **Segment:** Um serviço não confirmado iniciado pelo servidor, durante uma operação present. Trata-se da segmentação de um result-set.
Result-set-delete	**Delete:** Um serviço confirmado iniciado por um cliente para operação delete. A qual permite ao cliente a exclusão de um ou mais result-set.
Browse	**Scan:** Um serviço confirmado invocado pelo cliente para iniciar uma operação scan, que permite ao cliente avaliar/examinar um result set.
Sort	Possui dois serviços: • **Sort:** Um serviço confirmado invocado pelo cliente para a operação de Sort, que permite ao cliente classificar um result set. • **Duplicate Detection**: Um serviço confirmado invocado pelo cliente para iniciar uma operação de detecção de duplicidade no result-set.
Access Control	**Acess-control:** Um serviço confirmado iniciado pelo servidor e enviado ao cliente. É utilizado para validação de um cliente. Ele não inicia uma operação, e pode ser parte de uma operação já iniciada.
Account/ Resource Control	Possui três serviços: • **Resource-control:** Um serviço condicionalmente confirmado, iniciado pelo servidor e enviado ao cliente. É utilizado para notificar o cliente caso uma operação exceda um limite de recurso(disco, memória, etc). Pode ser parte de uma operação já iniciada. • **Trigger-resource-control:** Um serviço não confirmado iniciado pelo cliente. É usado para solicitar ao servidor que inicie o serviço Resource-control. • **Resource-report:** Um serviço confirmado iniciado pelo cliente. É usado pelo cliente para solicitar ao servidor um relatório de recursos consumidos.
Explain	Não inclui nenhum serviço, contudo utiliza os serviços Search e Retrieval. É usado para obter detalhes de implementação do servidor : nome do banco, atributos de tabelas, diagnósticos, estrutura do registro e especificação de elementos (colunas de uma tabela).
Extended Services	Um serviço confirmado, acionado pelo cliente para iniciar uma operação de serviço extendido.
Termination	**Close :** Um serviço confirmado iniciado pelo cliente ou servidor. Ele permite que o servidor ou cliente interrompa abruptamente todas as operações e sinaliza o termino de uma Z-Association.

5.4 Ações correspondentes a inicialização de uma Sessão Z39.50

A aplicação cliente envia um comando “OPEN”. O Z-client utiliza os parâmetros fornecidos no comando “OPEN” para abrir uma conexão TCP, formula uma APDU (Application protocol data Units) denominada “InitializeRequest” , transmite esta APDU para o servidor. Aguarda uma APDU denominada “InitializeResponse” do Z-server. O Z-Server, por sua vez cria uma Z-Association. Estas operações são transparentes a camada de aplicação, que vê somente o comando “OPEN” e a resposta.

5.5 Serviço Search

O serviço *Search* permite que um cliente submeta uma query a um banco de dados no servidor Z39.50. O resultado desta solicitação de consulta é um conjunto de registros denominado *result set.* Um *result set* pode ser utilizado para uma consulta posterior e assim formar um novo *result set*.

5.5.1 Parâmetros do Serviço Search

Os parâmetros do serviço *Search* são apresentados no quadro 2.

Quadro 2. Parâmetros do Serviço *Search*

Parâmetro	Pedido	Resposta
Query-type	**x**	
Query	**x**	
Database-names	**x**	
Result-set-name	**x**	
Replace-indicator	**x**	
Small-set-element-set-name	**opcional**	
Medium-set-element-set-names	**opcional**	
Preferred-record-syntax	**opcional**	
Small-set-upper-bound	**x**	
Large-set-lower-bound	**x**	
Medium-set-present-number	**x**	
Response-records		**X**
Result-count		**X**
Number-of-records-returned		**X**
Next-result-set-position		**X**
Search-status		**X**
Result-set-status		**X**
Present-status		**X**
Additional-search-information	**Opcional**	**Opcional**
Other-information	**Opcional**	**Opcional**
Reference-id	**Opcional**	**X**

O parâmetro *query-type* indica o tipo da query, ou seja, sua sintaxe. O padrão define seis tipos:

- *Type-0* usado quando o cliente e o servidor Z39.50 estabelecem uma sintaxe própria alheia ao padrão.
- *Type-1* é a notação polonesa reversa (RPN).
- *Type-2* é definida pela ISO8777
- *Type-100* é definida pelo padrão Z39.58
- *Type-101* é o type-1 extendido

O parâmetro *query* é uma string contendo uma query que obedece a sintaxe definida pelo seu tipo (*query-type).* Trata-se de uma consulta que utiliza uma ou mais clausulas de pontos de acesso ligados por conectores lógicos (*and*, *or*, e *and not*). Por exemplo, em uma base denominada “Historia Moderna” encontre os registros, cujo

ponto de acesso título contenha a palavra “guerra” e o ponto de acesso autor seja “Heródoto” (aqui observamos dois pontos de acesso ligados por um conector lógico “and”).

No escopo deste trabalho, será utlizado o tipo 1 ou RPN-Query (devido a sua maciça utilização pelos usuários do padrão), assim vamos descrever sua sintaxe:

@attrset X (@O) @attr Y=Z A (@attr Y=Z A @attr Y=Z A …)

Onde:

- **X** identifica o conjunto do atributo (será utilizado o Bib-1)
- **O** são os operadores lógicos (and, or, and-not, prox)
- **Y** Indica o tipo do atributo, que para Bib-1 assume os valores:
- 1 Use, 2 Relation, 3 Position, 4 Structure, 5 Truncation, 6 Completeness.
- **Z** Indica o valor do atributo conforme o seu tipo Y.
- **A** Indica o argumento de pesquisa da query.
- **( )** Indica que os termos são opcionais.

A seguir serão apresentados alguns exemplos de RPN-Query:

Neste exemplo a seguir podemos ver uma query que indica o conjunto do atributo como Bib-1, O tipo do atributo 1 Use, O valor do atributo 4 indica o acess point “titulo”. Pesquisa as obras que tem o título “Z39.50 Protocol”.

```
@attrset Bib-1 @attr 1=4 “Z39.50 Protocol”
```

No exemplo a seguir temos a mesma query do exemplo anterior, só que, O tipo do atributo 2 Relation, O valor do atributo 102 indica uma ordenação pela relevância.

```
@attrset Bib-1 @attr 1=4 @attr 2=102 “Z39.50 Protocol”
```

No exemplo a seguir vemos uma query com dois acess points: titulo e autor.

```
@attrset Bib-1 @and @attr 1=4 “Z39.50 Protocol” @attr 1=1003 “MIKE”
```

Cada clausula de ponto de acesso é constituída por um argumento de pesquisa e um atributo. O atributo qualifica o argumento de pesquisa. Um atributo corresponde a um ponto de acesso, o qual será comparado com o argumento de pesquisa fornecido para localizar a informação desejada. Cada atributo é formado pelo par tipo do atributo e valor do atributo. Onde o tipo do atributo está associado a um conjunto de atributos (o padrão define 6 conjuntos: Bib-1, Exp-1, Ext-1, CCL-1, GILS e STAS), cada conjunto de atributos define um conjunto de tipos de atributos e seus valores.

5.6 O Result Set

Após o comando de pesquisa search, o *result set* está disponível no servidor para que o cliente tenha acesso ao seu conteúdo. Associado com a base de dados, existe um o mais esquemas. Um esquema representa um entendimento comum entre o cliente e o servidor sobre as informações contidas nos registros de uma base de dados, que permite a seleção de parte da informação através da especificação de um elemento.

O esquema define uma estrutura de registro abstrata, que uma vez aplicada aos registros da base de dados resulta em um registro abstrato, este representa uma visão particular de um registro da base de dados. Comparando com uma base de dados relacional, o esquema poderia ser entendido como uma view (uma tabela virtual que contém colunas de uma ou mais tabelas reais da base de dados) e o registro abstrato seria formado pelas colunas da view.

O servidor Z39.50 gera um registro que seja exportável (entendido pelo cliente Z39.50) de acordo com o que foi definido no registro abstrato.

O result set é constituído por uma lista ordenada de itens. Cada item aponta para um registro da base de dados. Ele é usado pelo mecanismo de seleção para

transferir os registros da base de dados, identificados por uma query, para o cliente. O result set é na verdade uma referência aos registros propriamente ditos.

O modelo abstrato, sugerido pelo padrão, define o result set como um vetor ou um conjunto de itens que esta associado a uma base de dados e referencia os registros desta base. Cada item do result set é formado por uma tripla, constituída de : um número que representa a ordem do item no result set, o nome da base de dados, e um identificador único de um registro dentro da base de dados.

Os registros em um result set não são necessariamente ordenados por nenhum elemento. Contudo, existe o serviço “Sort”, que no caso de ser implementado pelo servidor, permite uma ordenação do result set por um determinado elemento do mesmo.

5.7 Implementações de Servidores Z39.50

O protocolo Z39.50 foi implementado através dos kits : YAZ toolkit (Hammer, 2000) e JZKIT (Ibbo, 2000). Tais sistemas suportam as funcionalidades de busca e recuperação previstas no protocolo Z39.50. O YAZ foi desenvolvido na linguagem C e disponibiliza seus fontes para que possam ser adaptados a biblioteca digital com a qual ira interagir. O JZKIT foi escrito na linguagem JAVA e igualmente disponibiliza seus fontes para adaptação ao contexto que irá operar. Há contudo, um elemento complicador no processo de adaptação destes sistemas: o estabelecimento da comunicação entre o ZSERVER e a base de dados. Isto ocorre por que o Z39.50 visualiza uma base de dados como um modelo genérico. Assim as implementações do Z39.50 aqui mencionadas exigem um considerável esforço de programação para que seja possível o funcionamento pleno do serviço de recuperação de informação proporcionado pelo Z39.50.

5.8 Conclusão

O padrão Z39.50 se apresenta como uma solução de fato para prover o intercâmbio de informações entre bibliotecas digitais. Através de uma arquitetura que

compreende serviços, protocolo e uma camada de abstração, o padrão Z39.50 consegue ser versátil ao ponto de se adaptar facilmente a sistemas heterogêneos, é precisamente nesta característica que reside sua ampla utilização no escopo das bibliotecas digitais.

Existem algumas soluções para adaptar bibliotecas digitais proprietárias ao protocolo Z39.50, mas a atividade de instalação destas soluções é complexa, requerendo até mesmo a recodificação. No próximo capítulo é apresentada uma solução a este problema.

Capítulo 6. A Linguagem DCM-BIB

Infelizmente, nem todas as bibliotecas digitais estão em conformidade com os padrões de interoperabilidade em bibliotecas digitais (Dublin Core, Z39.50 ou Open Archives), por isso, recebem a classificação de proprietárias. Tal fato implica na redução da visibilidade do acervo disponibilizado por estas bibliotecas, visto que a consulta ao acervo destas bibliotecas só pode ser realizado pelo sistema de busca proprietário das mesmas, e sempre individualmente. A inserção destas bibliotecas em uma busca integrada (uma mesma formulação de busca aplicada a diversas bibliotecas) não é possível.

Várias iniciativas, como JZKIT (Ibbo, 2000), DBISERVER (Taylor, 2002), e o YAZ (Hammer, 2000), oferecem implementações de servidores Z39.50 para bibliotecas digitais. Contudo, a atividade de adaptação destas soluções a uma biblioteca proprietária é complexa, envolvendo inclusive atividades de recodificação do servidor de modo a adequar ao esquema do banco de dados onde reside o acervo de uma biblioteca digital proprietária.

Este capítulo apresenta a linguagem baseada em XML para a descrição da relação entre metadados de bibliotecas digitais proprietárias e metadados Dublin Core proposta nesta dissertação. Esta linguagem, chamada de DCM-BIB (Descrição da Conversão de Metadados para Bibliotecas Digitais), tem por objetivo facilitar a adaptação de servidores padrões de bibliotecas digitais, como os servidores Z39.50 e provedores de dados OAI, tornando-os capazes de processar comandos de busca com argumentos de pesquisa que contenham metadados Dublin Core. A DCM-BIB poderá ser usada pelos administradores das bibliotecas digitais devido a sua simplicidade e praticidade. Além da linguagem, este capítulo apresenta uma interface web possibilitando gerar o arquivo de relação, facilitando assim a atividade de especificação das relações entre metadados proprietários em metadados Dublin Core.

Apesar de poder ser usada em qualquer servidor que adote os metadados Dublin Core, esta dissertação trata especificamente de servidores Z30.50. Nós definimos um procedimento de utilização do arquivo de relação para a conversão da primitiva de busca Z39.50 para comandos SQL adequados ao banco de dados da biblioteca digital.

A linguagem DCM-BIB e o procedimento de conversão entre consultas Z39.50 em consultas SQL permitirão o desenvolvimento de servidores Z39.50 que sejam de fácil integração a qualquer tipo de biblioteca digital baseada em banco de dados relacionais. Adotando tal tipo de servidor, um administrador da biblioteca proprietária poderá simplesmente criar um “arquivo de relação”. Este arquivo especifica a relação entre os metadados proprietários e os metadados Dublin Core usando a linguagem DCM-BIB. Na partida do servidor, o mesmo acessaria o arquivo de relação de maneira a converter as consultas Z39.50 em consultas SQL.

Para testar a linguagem DCM-BIB e a eficiência do algoritmo de mapeamento de metadados Dublin Core para os metadados proprietários, foi implementado um protótipo com uma interface gráfica que permitiu que fossem efetuados vários ensaios (formulação de consultas do Z39.50 e obtenção do correspondente em SQL, com base no mapeamento definido através da DCM-BIB) contra uma base de dados, que representa um suposto acervo de uma biblioteca digital proprietária.

6.1 A linguagem DCM-BIB

A linguagem DCM-BIB (Descritivo de conversão de metadados para bibliotecas digitais) é uma descrição do mapeamento entre metados Dublin Core e os metadados proprietários de uma dada biblioteca digital (que não utiliza o Dublin Core). A partir dos elementos da linguagem, pode-se representar em um documento XML a correlação dos metadados e também o relacionamento entre as tabelas que compõe a base de dados relacional usada para armazenar o acervo digital de uma biblioteca proprietária. A partir de um número reduzido de elementos, a linguagem pretende ser simples, objetiva, e não requer que o administrador da biblioteca digital tenha habilidades de programador para descrever a correlação entre os metadados.

6.1.1 Estrutura básica da DCM

Como HTML, uma especificação DCM é dividida em duas seções: o cabeçalho (*head*) e o corpo (*body*). Ambas precisam estar incluídas dentro do *tag* <dcm>. A Figura 6 apresenta a estrutura básica de um documento DCM. O cabeçalho contém informações gerais relativas ao documento especificados por um conjunto de metadados. O corpo contém a definição dos relacionamentos entre metadados Dublin Core e metadados proprietários.

```
<dcm>
   <head>
      <!-- Definição de atributos -->
   </head>
   <body>
      <!-- Definição das relações entre metadados -->
   </body>
</dcm>
```

Figura 6. Estrutura básica de um código DCM

6.1.2 Cabeçalho de um documento DCM

O cabeçalho contém a definição de informações gerais relativas ao documento. Estas definições são realizadas com o uso do elemento *meta*. Cada elemento *meta* especifica um par propriedade/valor nos atributos *name* e *content*, respectivamente:

- *name=string*: este atributo especifica o nome de atributos gerais da biblioteca digital, como título, nome do responsável, URL, etc.

- content=string: este atributo descreve o conteúdo do atributo definido em *name*.

Por exemplo, na Figura 7 são usadas uma série de elementos meta.

```
<dcm>
   <head>
      <meta name="title" content="Biblioteca Digital de Literatura " />
      <meta name="base"
         content="http://www.sidie.nurcad.ufsc.br/bdnupill" />
      <meta name=="responsable" content="Roberto Willrich" />
   </head>
   <body>
      <!-- Definição das relações entre metadados -->
   </body>
</dcm>
```

Figura 7. Representação dos elementos Meta

6.1.3 Corpo de um documento DCM

No corpo de um documento DCM são definidas as relações entre os metadados proprietários da biblioteca digital e os metadados Dublin Core. Além disso, são definidos os relacionamentos entre as tabelas do banco de dados onde o acervo digital se encontra. Estes relacionamentos são importantes para o processo de conversão de consultas Z39.50 em SQL.

O corpo de um documento DCM é dividido em duas seções, como ilustra a Figura 8, na seção *dcm:mapping* estabelece-se a relação entre os metadados proprietários e Dublin Core; a seção *dcm:relations* descreve o relacionamento entre as tabelas que compõe o banco de dados da biblioteca digital.

```
<dcm>
  <head>
    <!-- Definição de atributos -->
  </head>
  <body>

    <dcm.mapping>
      <!-- Definição de relação entre metadados -->
      <dc-title>          </dc-:title>
      <dc-creator>        </dc-creator>
      <dc-subject>        </dc-subject>
      <dc-source>         </dc-source>
      <dc-description>    </dc-description>
      <dc-publisher>      </dc-publisher>
      <dc-contributors>   </dc-contributors>
      <dc-date>           </dc-date>
      <dc-type>           </dc-type>
      <dc-format>         </dc-format>
      <dc-identifier>     </dc-identifier>
      <dc-language>       </dc-language>
      <dc-relation>       </dc-relation>
      <dc-coverage>       </dc-coverage>
      <dc-rights>         </dc-rights>
    </dcm.mapping>

    <dcm.relations>
      <!-- Definição das relações do banco de dados -->
      <dcm:relation>      </dcm:relation>
      <dcm:relation>      </dcm:relation>
      ...
    </dcm.relations>
```

```
  </body>
</dcm>
```

Figura 8. Corpo de um documento DCM

6.1.4 Elemento dcm.mapping

Esta parte do documento DCM especifica o mapeamento um-a-um entre os metadados proprietários usados pela biblioteca digital e os metadados DC. Nesta parte é definida uma relação para cada metadado DC. A figura 15 ilustra a especificação dos 15 metadados Dublin Core. Contudo, nesta parte estarão presentes apenas os metadados DC que tenham efetivamente uma correlação com os metadados proprietário que estão registrados no banco de dados.

Para especificação de tal relação foi definido um conjunto de elementos:

- **table**: descreve o nome da tabela do banco de dados que contém o metadado DC.
- **field:** descreve o nome de uma coluna pertencente a tabela descrita em **table**.
- **type:** indica o tipo do dado (conforme definido na tabela do banco de dados)

A Figura 9 ilustra o uso destes elementos. Neste exemplo, o valor de dc-title pode ser encontrado no campo *Titulo* da tabela *Documento*, e cujo tipo de dados é Varchar(100).

```
<dc-title>
   <table> Documento </table>
   <field> Titulo </field>
   <type> VarChar(100) </type>
</dc-title>
```

Figura 9. Exemplo de definição de um mapeamento

6.1.5 Elemento dcm.relations

Esta parte do documento DCM especifica somente os relacionamentos entre as tabelas do banco de dados que estejam associados aos metadados DC (conforme

definido na seção *mapping)*. Cada relacionamento é definido pelo elemento *dcm:relation*. Para a especificação de tal relação foi definido um conjunto de elementos:

- **source**: descreve o nome da tabela primária, ou seja, a tabela de onde se origina o relacionamento. Este elemento tem os parâmetros *table*, que define o nome da tabela e *pk* que descreve o nome da chave primária da tabela.

- **target**: descreve o nome da tabela com a qual a tabela descrita em **source** se relaciona através da igualdade **pk = fk**. Este elemento tem os parâmetros *table*, que define o nome da tabela e *fk* que descreve o nome da chave estrangeira da tabela descrita em **source** e presente em **target.**

A Figura 10 mostra um modelo de entidade relacionamento onde um autor possui uma ou mais obras e uma obra pode estar associada a um ou mais autores. Logo em seguida a representação em DCM-BIB deste relacionamento.

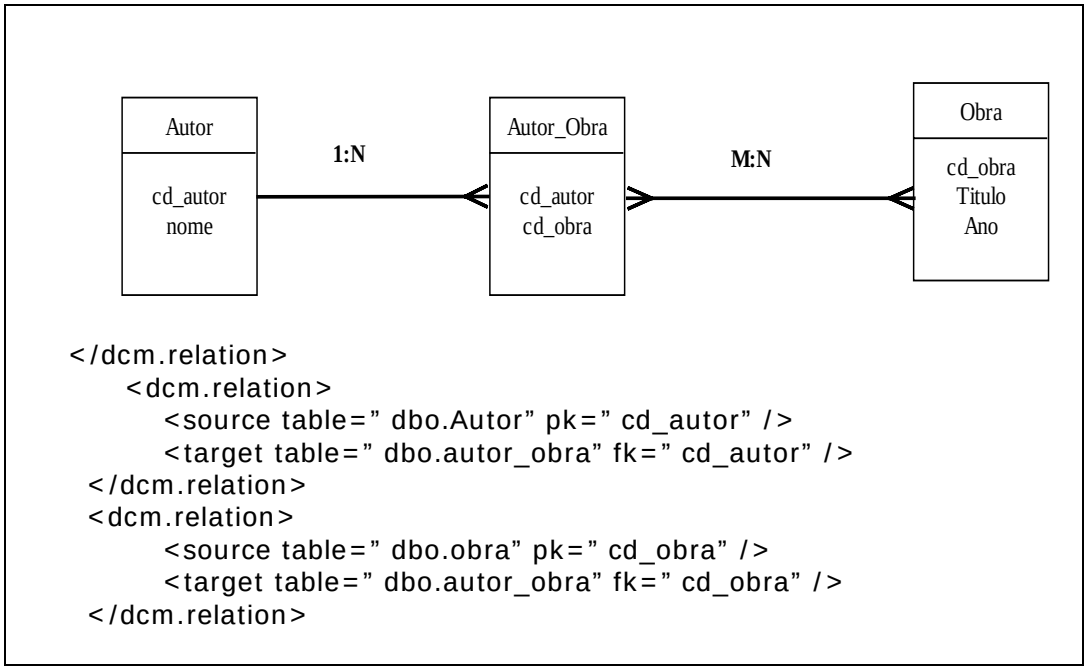


Figura 10. Modelo de Entidade relacionamento e sua descrição em DCM-BIB

Note que na DCM-BIB, as relações entre as tabelas são descritas em pares. Os relacionamentos transitivos são inferidos implicitamente, ou seja, se a tabela A se relaciona com a tabela B e a tabela B com a C então A e C possuem uma ligação através

de B. As chaves primárias compostas (com mais de uma coluna) também podem ser descritas pela repetição dos elementos pk e fk. Os Elementos pk de uma dada tabela devem ter o seu correspondente fk na tabela relacionada.

6.1.6 Esquema XML para o documento DCM-BIB/XML

A Figura 11 apresenta o esquema XML da linguagem DCM-BIB. Este foi confirmado como correto pelo validador XSV 2.8-1 da W3C (W3C 2001b).

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 targetNamespace= "http://www.pgcc.inf.ufsc.br"
 xmlns="http://www.pgcc.inf.ufsc.br"
 elementFormDefault="qualified">

<xs:complexType name="maptype">
    <xs:sequence>
       <xs:element name = "Table" type="xs:string"/>
       <xs:element name = "Field" type="xs:string"/>
       <xs:element name = "Type" type="xs:string"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="sourcetype">
   <xs:attribute name = "table" type="xs:string"/>
   <xs:attribute name = "pk"   type="xs:string"/>
</xs:complexType>

<xs:complexType name="targettype">
   <xs:attribute name = "table" type="xs:string"/>
   <xs:attribute name = "fk"   type="xs:string"/>
</xs:complexType>


<xs:complexType name="relationtype">
    <xs:sequence>
       <xs:element name = "source"    type="sourcetype"/>
       <xs:element name = "target"    type="targettype"/>
   </xs:sequence>
</xs:complexType>



<xs:complexType name="dublincoretype">
       <xs:sequence>
          <xs:element name = "DC-Title" type="maptype"
                      minOccurs="0" maxOccurs="1"/>
           <xs:element name = "DC-Creator" type="maptype"
                      minOccurs="0" maxOccurs="1"/>
           <xs:element name = "DC-Subject" type="maptype"
                      minOccurs="0" maxOccurs="1"/>
```

```
            <xs:element name = "DC-Description"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-Publisher"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-Date"  type="xs:date"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-ResourceType"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-ResourceIdentifier"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-Language"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-OtherContributor"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-Format"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
            <xs:element name = "DC-RightsManagement"  type="maptype"
                        minOccurs="0"  maxOccurs="1"/>
        </xs:sequence>
</xs:complexType>

<xs:complexType  name="metatype">
    <xs:attribute  name = "meta"     type="xs:string"/>
    <xs:attribute  name = "content"   type="xs:string"/>
</xs:complexType>


<xs:complexType name="bodytype">
        <xs:sequence>
         <xs:element name = "dcm.mapping" type="dublincoretype"/>
         <xs:element name = "dcm.relations" type="relationtype"/>
        </xs:sequence>
</xs:complexType>


<xs:element name="dcm">
    <xs:complexType>
       <xs:sequence>
          <xs:element name="head"  type= "metatype"/>
          <xs:element name="body"  type= "bodytype"/>
       </xs:sequence>
     </xs:complexType>
</xs:element>

</xs:schema>
```

Figura 11. Esquema da linguagem DCM

6.1.7 Exemplo de Documento DCM-BIB

Esta seção ilustra o uso da linguagem DCM-BIB a partir de um dado cenário de uso. Assim foi definida uma base de dados com três tabelas, que pretendem simular uma parte da base de dados de uma biblioteca digital. A Figura 12 apresenta o modelo de entidade relacionamento desta base.

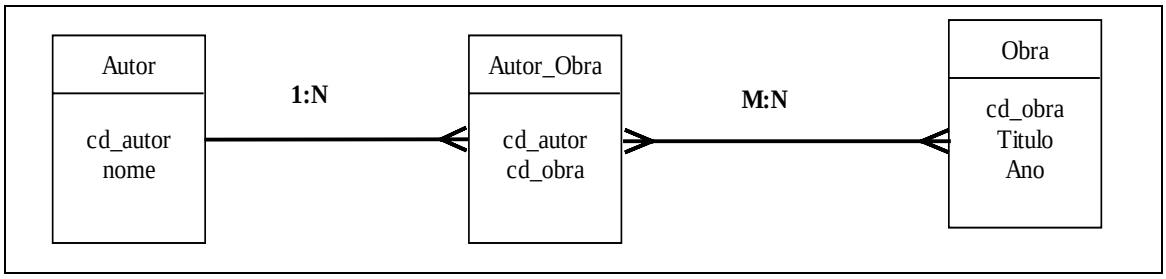


Figura 12. Modelo de entidade relacionamento da Biblioteca Digital de teste

Para o esquema de banco de dados definido na Figura 12 teríamos o documento DCM-BIB apresentado na Figura 13.

```
<dcm>
  <head>
    <meta name="Title" content="Biblioteca Digital Teste" />
    <meta name="Responsible" content="Fernando Ponce de Leon" />
  </head>

  <body>
    <dcm-mapping>
      <dc:title>
        <table> obra</table>
        <field> Titulo </field>
        <type> VarChar(140) </type>
      </dc:title>
      <dc:creator>
        <table> autor</table>
        <field> nome </field>
        <type> VarChar(40) </type>
      </dc:creator>
    </dcm-mapping>

    <dcm-relations>
      <dc:relation>
          <source name="autor" pk="cd_autor" />
           <target name="autor_obra" fk="cd_autor" />
      </dc:relation>
      <dc:relation>
```

```
            <source name=" obra" pk=" cd_obra" />
             <target name=" autor_obra" fk=" cd_obra" />
        </dc:relation>
    </dcm-relations>
  </body>
</dcm>
```

Figura 13. Exemplo de um documento DCM-BIB

6.2 Procedimento de conversão de consultas Z39.50 em SQL

Esta seção apresenta o procedimento de tradução de uma query formulada por um client Z39.50 para o correspondente na linguagem SQL, chamado aqui de DCM-Z3950-SQL. Ele vai funcionar como um wrapper ou tradutor.

O DCM-Z3950-SQL poderá ser utilizado por servidores Z39.50 já existentes. Ou seja, no código fonte do servidor Z39.50 poderá ser incluída uma chamada ao modulo DCM-Z3950-SQL, que por sua vez, converte uma query formulada pelo Z-Client em um comando SQL-ANSI (obedecendo a correlação de metados descrita pela DCM-BIB), e o devolve para o Servidor Z39.50, que por fim submete esta consulta SQL a base de dados.

6.2.1 Leitura do arquivo DCM-BIB

O DCM-Z3950-SQL inicialmente lê o documento DCM-BIB, interpreta a descrição da relação dos metadados e as eventuais relações entre as tabelas do banco de dados e as armazena em duas tabelas, que chamaremos de *Mapping* e *Relations*.

Em cada linha ou ocorrência da tabela *Mapping* serão armazenados os seguintes dados (obtidos da seção *dcm:mapping* do documento DCM-BIB):

- Elemento DC: Nome do elemento Dublin Core;
- Table: Nome da tabela onde o dado representado pelo Elemento Dublin Core reside;
- Field: Nome da coluna que armazena o dado associado ao elemento DC;

- Type: Tipo do dado;
- Alias: Uma letra que representa o nome da tabela.

Em cada linha ou ocorrência da tabela *Relations* serão armazenados os seguintes dados (obtidos da seção *dcm:relations* do documento DCM-BIB) :

- Source: Nome da tabela que origina o relacionamento;
- Pk: Nome da primary key de Source;
- Target: Nome da tabela que se relaciona com Source;
- Fk: Nome da foreing key que se relaciona com Pk.

6.2.2 Tradução da Consulta Z39.50 para SQL

A interação do wrapper com o servidor Z39.50 é mostrado na Figura 14. As operações no diagrama de colaboração são a abaixo descritas:

1. O administrador da biblioteca digital ativa o Z39.50 Server, acionando o método iniciar.
2. O Z39.50 Server aciona o método “Carregardocumento” do DCM-Z3950-SQL, este ultimo lê e interpreta o documento DCM-BIB (conforme apresentado na seção 6.2.1).
3. O Usuário submete uma consulta através do comando de busca do protocolo Z39.50 que contém argumento Dublin Core.
4. O cliente Z39.50 aciona o método “SendSearch” do Z39.50 Server

5. O Servidor Z39.50 aciona o método “Traduzir” do Wrapper DCM-Z3950-SQL.

6. O Servidor Z39.50, de posse do comando SQL, devidamente convertido pelo método “Traduzir”, aciona o método SendCmd para o banco de dados relacional, e disponibiliza o result set obtido para o Cliente Z39.50.

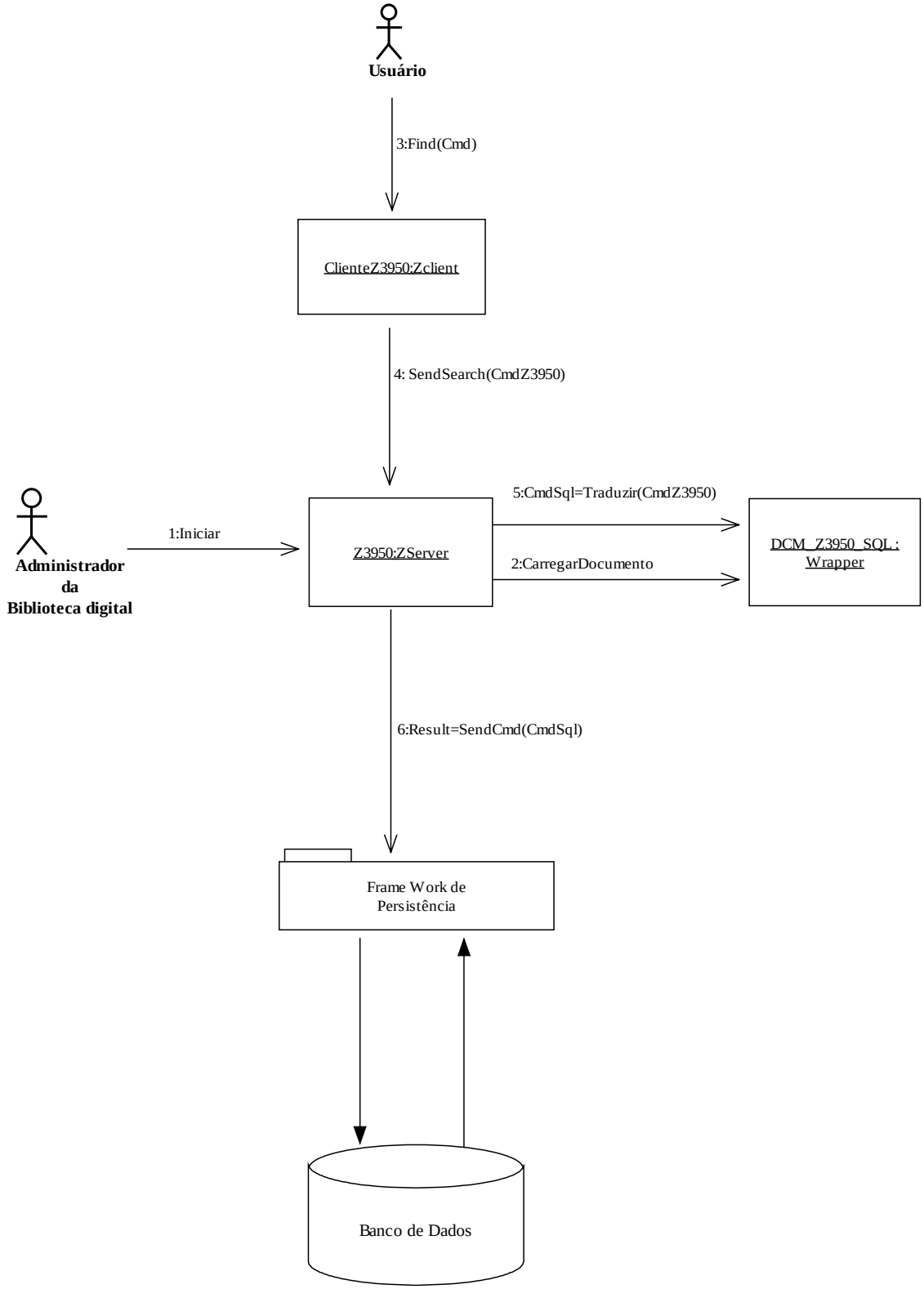


Figura 14. Interação do Wrapper DCM-Z3950-SQL com o Servidor Z39.50

A seguir será mostrado um pseudo-código do Wrapper DCM-Z3950-SQL:

```
/* Definição de tipos de dados
Type mapping
    Begin
        Elemento_DC : String;
        Tabela      : String;
        Campo       : String;
        Tipo        : String;
        Formato     : String;
        Alias       : String;
    End
 Type relation
    Begin
       Tabela1     : String;
       PK          : String;
       Tabela2     : String;
       FK          : String;
    End;
Type  T3_TABELA_RELACAO
     Begin
        Tabela     : String;
        Alias      : String;
    End;



 /* vetor que armazena  dados obtidos no bloco <mapping> do documento
DCM-BIB
  V1_DC_PROP    :   array (1..15)    of mapping;

/* vetor que armazena os eventuais relacionamentos entre as tabelas, obtidos
no bloco  <relations> do documento DCM-BIB */
  V2_RELACAO    :   array (1..300)  of relations;

/* vetor necessário para relacionar a tabela do banco a um alias
  V3_TABELA_RELACAO  :   array (1..50)  of T3_TABELA_RELACAO;

/* Vetor usado para armazenar os operadores lógicos existentes na query tipo
1 que usa
    a notação polonesa reversa (RPN)
  PILHA1_OPERADORES : array (1..100)  of String;

/* Vetor usado para armazenar os operandos existentes na query tipo 1.
  FILA1_OPERANDOS      : array (1..100)  of String;

  BOTTOM_PILHA1 : Integer;
  TOP_FILA1 : Integer;
ConverteQueryTipo1 (Query:String)
    Var
       CmdQuery : String
       Palavra      : String
```

```
      TamanhoPalavra : Integer
      CmdSelect : String
   Begin
      CmdQuery = Query

      /*Esta rotina irá separar os argumentos e os operadores

      While  Length(CmdQuey) >  0  Do
            Begin
               Palavra  = ObtemPalavra (CmdQuery)
               TamanhoPalavra = Length(Palavra)

               If  PrimeiroCaracter(Palavra) = "@"  Then
                  DeleteCaracter(Palavra,1,1)  /*elimina o caracter da pos. 1
tamanho 1
                  If  (Palavra = "and")  Or
                     (Palavra = "or")   Or  (Palavra = " not and")  Then
                     CarregaPilha (Palavra)
                  ElseIf  (Palavra = "attr")  Then
                     DeleteCaracter(Palavra,1,4)  /*elimina o caracter  pos.
1 tamanho 4
                  EndIf
               Else
                  CarregaFila(Palavra)
               EndIf

               DeleteCaracter(CmdQuery,1,TamanhoPalavra))
            EndWhile

        /* Após a pilha e a fila terem sido carregadas com os operadores
lógicos e operandos da query tipo 1 é montado o comando SELECT */

        CmdSelect  =  "SELECT  "  + ObtemColunasSelect + " FROM  "  +
                        ObtemTabelasSelect  + ObtemClausulaWhere

        Return  With  CmdSelect
  End;
```

6.3 Interface gráfica para criar um documento DCM-BIB

Para facilitar ainda mais a atividade de descrever a correlação entre os metadados Dublin Core e os metadados proprietários, foi definida uma interface gráfica, que utiliza um formulário que é preenchido pelo Administrador da biblioteca digital para gerar automaticamente um documento XML com a descrição DCM-BIB. A Figura 15 mostra este formulário. Neste formulário, o administrador da Biblioteca digital informa no painel "mapeamento" as correlações entre os metadados Dublin Core e os metadados proprietários, informando a tabela e a coluna onde os mesmos se encontram

na base de dados. No painel “Relacionamento” são informados os relacionamentos entre as tabelas, ou seja, a tabela de origem e a tabela de destino com suas chaves primária (Pk) e estrangeira(Fk).

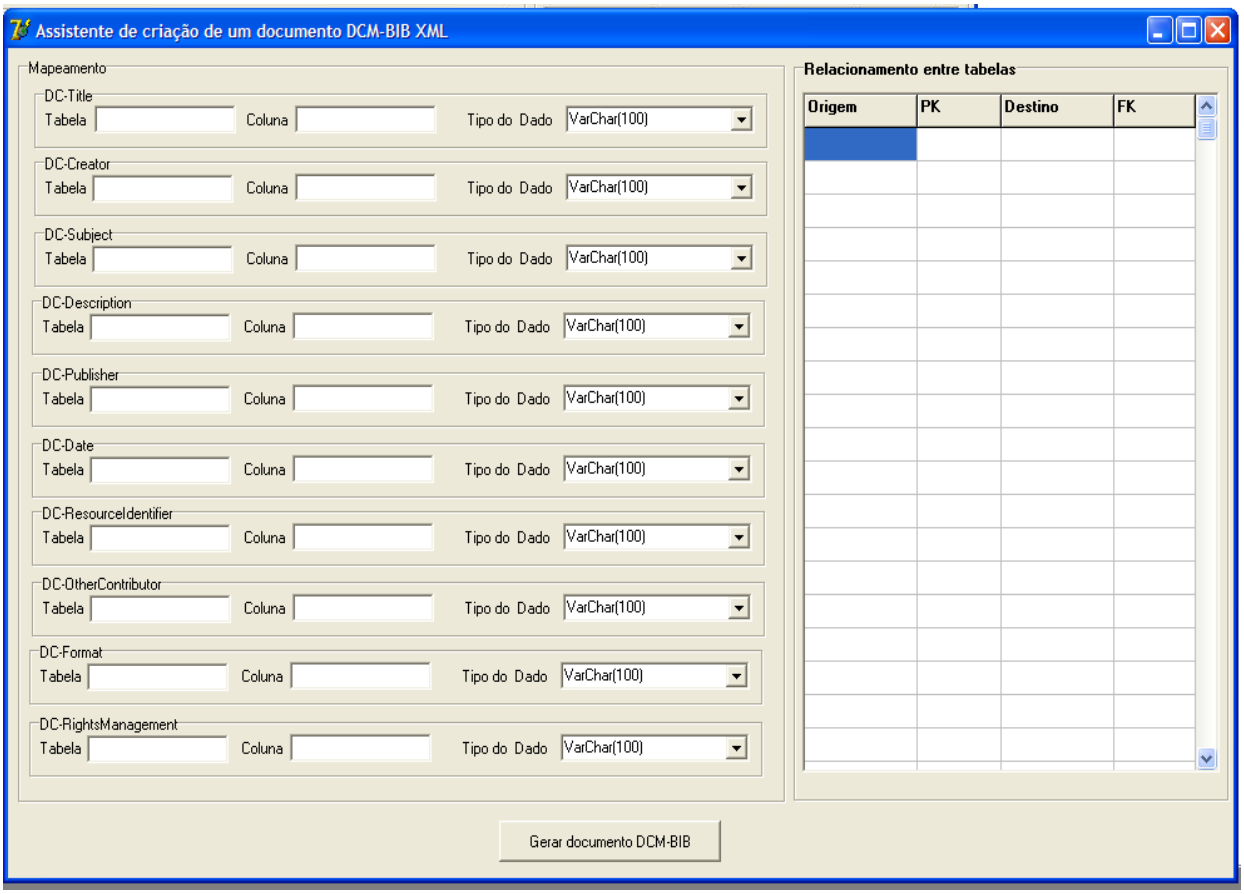


Figura 15. Interface para geração automática de um Documento DCM-BIB

6.4 Protótipo implementado

Com o objetivo de validar o algoritmo (apresentado na seção 6.2.2) usado no módulo DCM-Z3950-SQL para a tradução de queries Z39.50 em SQL, foi criado um ambiente de ensaio, que considera as tabelas da Biblioteca digital da UFSC, a BDNUPIL. Este ambiente utilizou um banco de dados relacional e uma ferramenta de desenvolvimento de aplicações. A figura 16 apresenta o modelo de entidade relacionamento da BDNUPIL.

MODELO DE ENTIDADE/RELACIONAMENTO
DA
BDNUPIL - UFSC

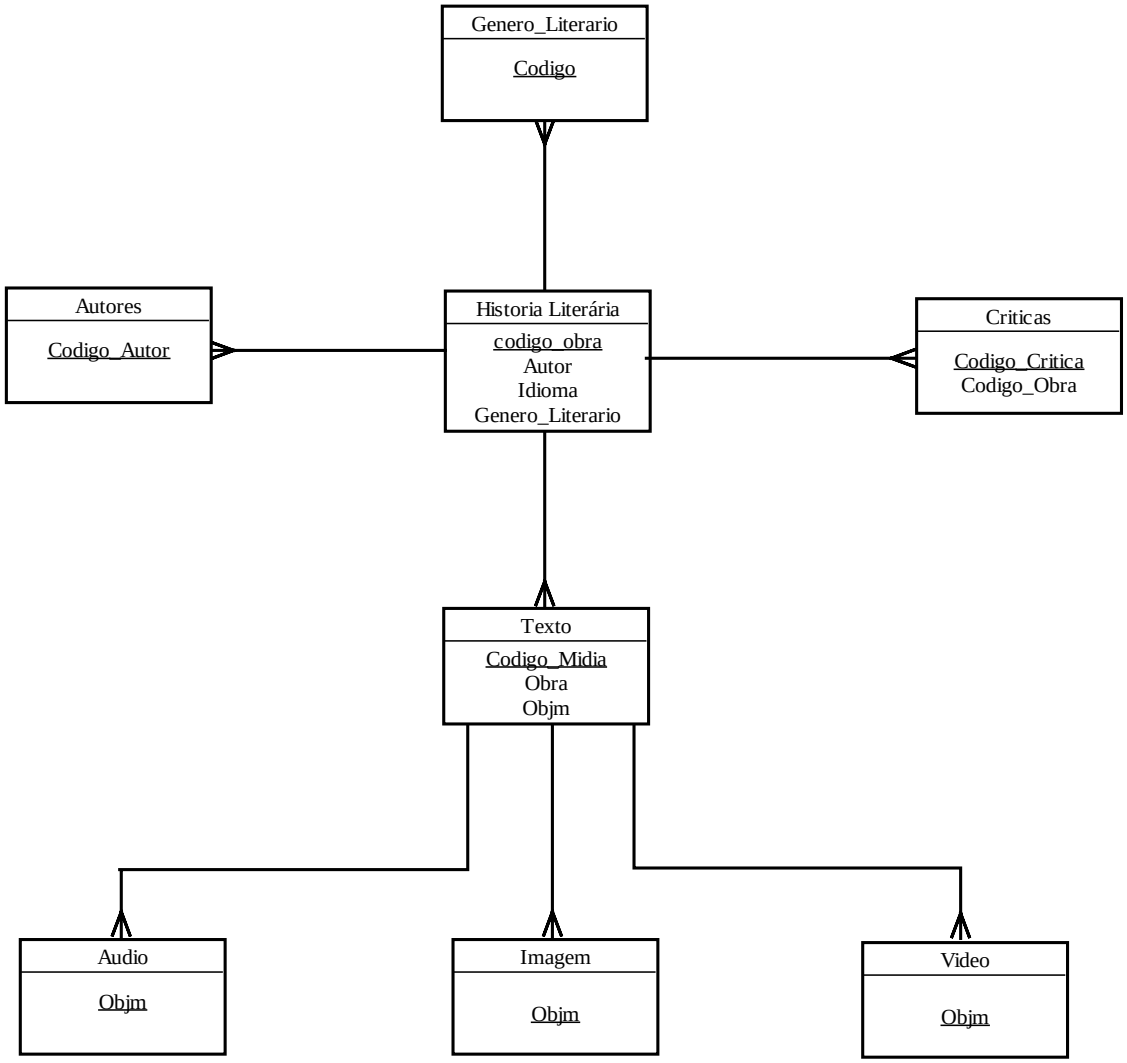


Figura 16. Modelo de Entidade/Relacionamento da BDNUPIL

A figura 17 mostra o mapeamento entre os metadados Dublin Core e os respectivos correspondentes no contexto da BDNUPIL, e também o relacionamento entre as tabelas.

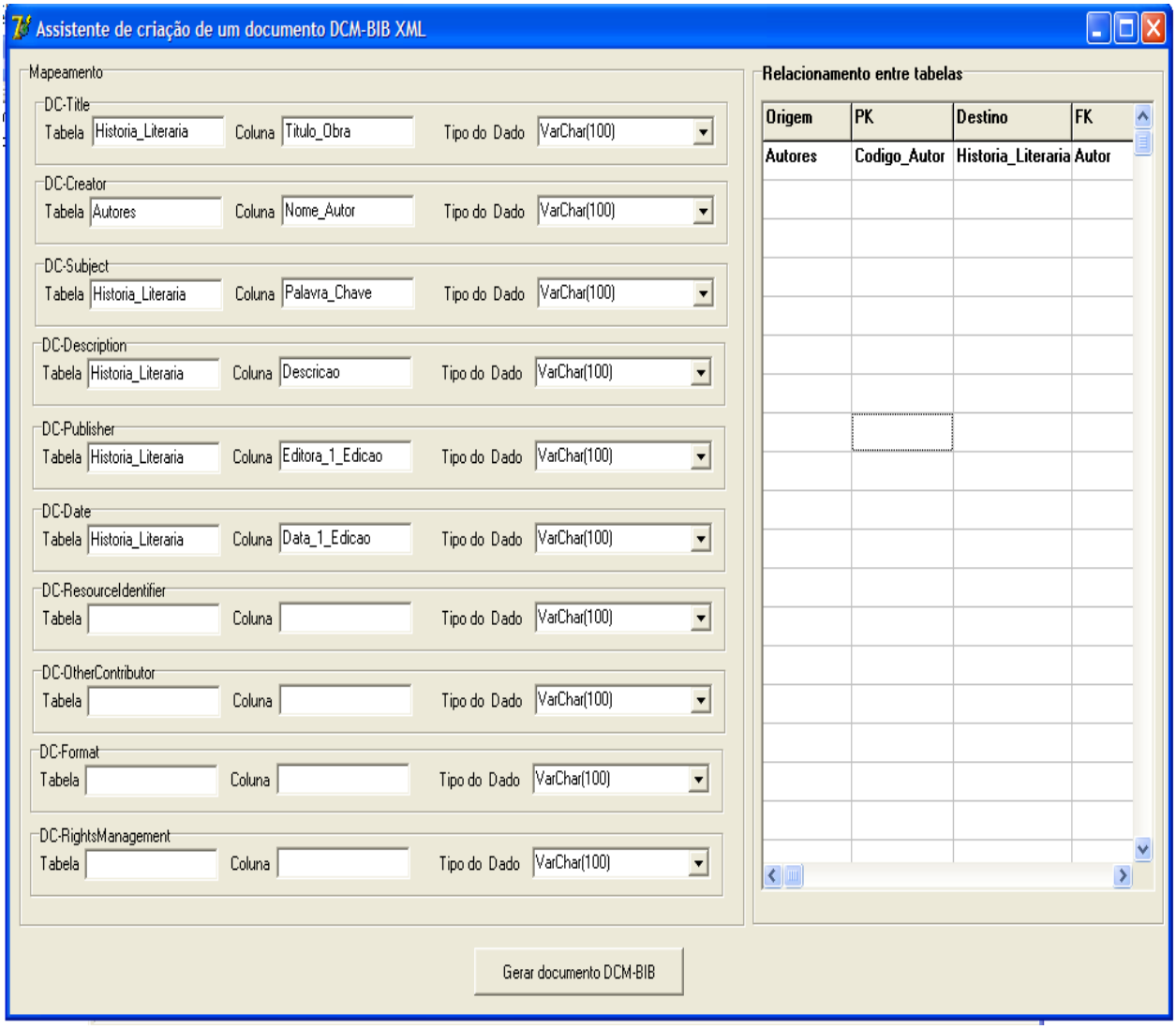


Figura 17. Mapeamento dos metadados DC e relacionamento entre as tabelas

A figura 18 apresenta o documento DCM-BIB gerado a partir das informações colhidas na interface gráfica apresentada na figura 17.

```
<dcm>
  <head>
    <meta name="Title" content="BDNUPIL" />
    <meta name="Responsible" content="Roberto Willrich" />
  </head>
  <body>
    <dcm-mapping>
      <dc:title>
        <table>Historia_Literaria</table>
        <field> Titulo_Obra </field>
        <type> VarChar(80) </type>
      </dc:title>
      <dc:subject>
        <table>Historia_Literaria</table>
        <field> Palavra_chave </field>
        <type> VarChar(80) </type>
      </dc:subject>
      <dc:description>
        <table>Historia_Literaria</table>
        <field> Descricao </field>
        <type> VarChar(80) </type>
      </dc:description>
      <dc:publisher>
        <table>Historia_Literaria</table>
        <field> Editora_1_Edicao</field>
        <type> VarChar(80) </type>
      </dc:publisher>
      <dc:date>
        <table>Historia_Literaria</table>
        <field> Data1_Edicao</field>
        <type> VarChar(4) </type>
      </dc:date>
      <dc:creator>
        <table> Autores</table>
        <field> nome_autor </field>
        <type> VarChar(60) </type>
      </dc:creator>
    </dcm-mapping>
    <dcm-relations>
      <dc:relation>
          <source name="Autores" pk="Codigo_Autor" />
           <target name="Historia_Literaria" fk="Autor" />
      </dc:relation>
    </dcm-relations>
  </body>
</dcm>
```

Figura 18. Documento DCM-BIB

A figura 19 mostra um teste de tradução de uma query tipo 1 (definida pelo Z39.50) em um comando SQL com sua posterior validação em uma ferramenta de acesso a base. Neste teste, obtem-se a relação de todos os autores cujo o nome inicia com a letra “A”. Para mensurar o tempo necessário para efetuar a tradução, foi incluído um indicador de tempo, que mostra quando o comando foi enviado e quando o mesmo foi retornado na sua forma traduzida em SQL. Podemos constatar, nos ensaios efetuados, que o tempo gasto para traduzir o camando search do Z39.50 em SQL é inferior a um segundo.

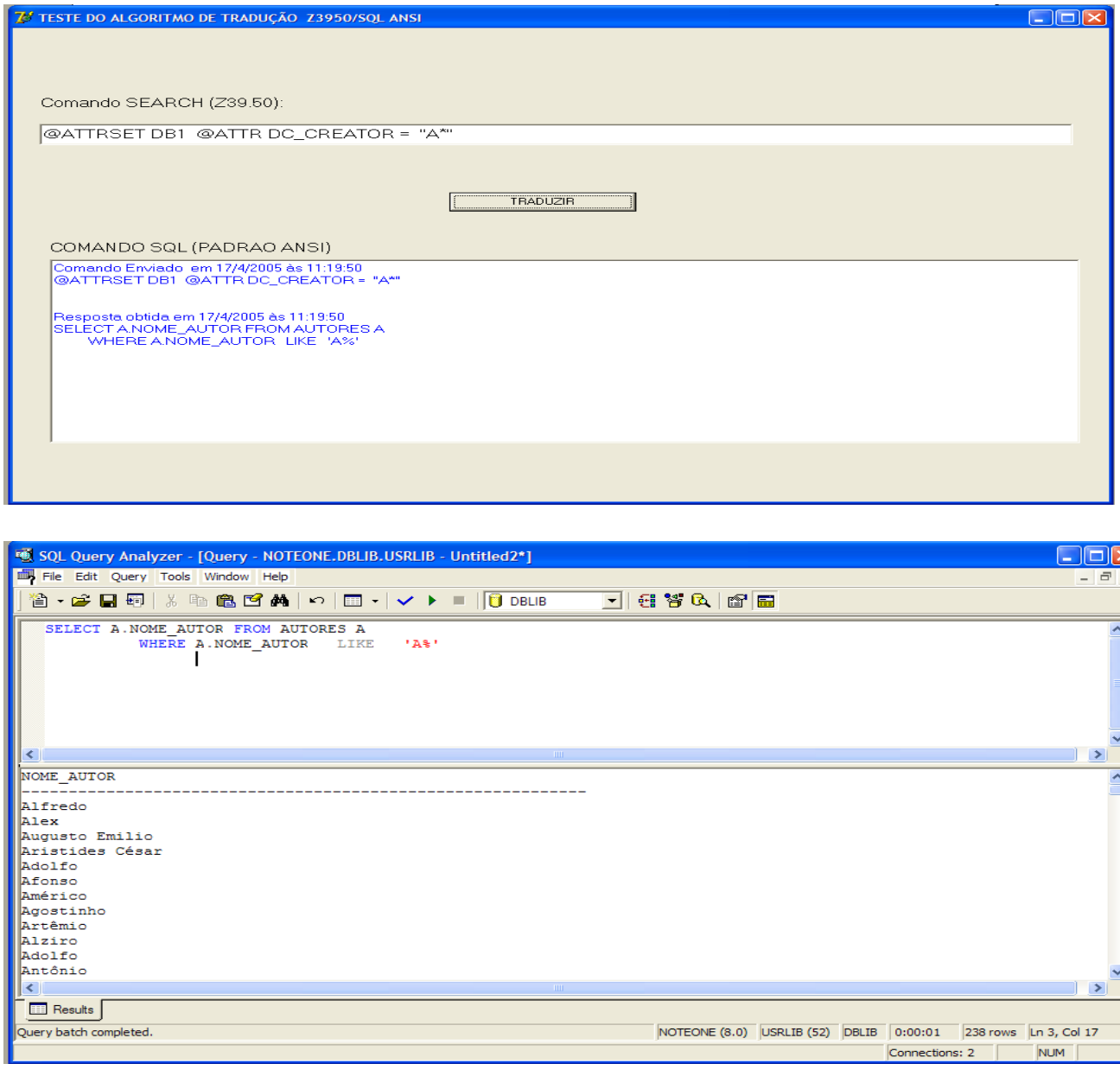
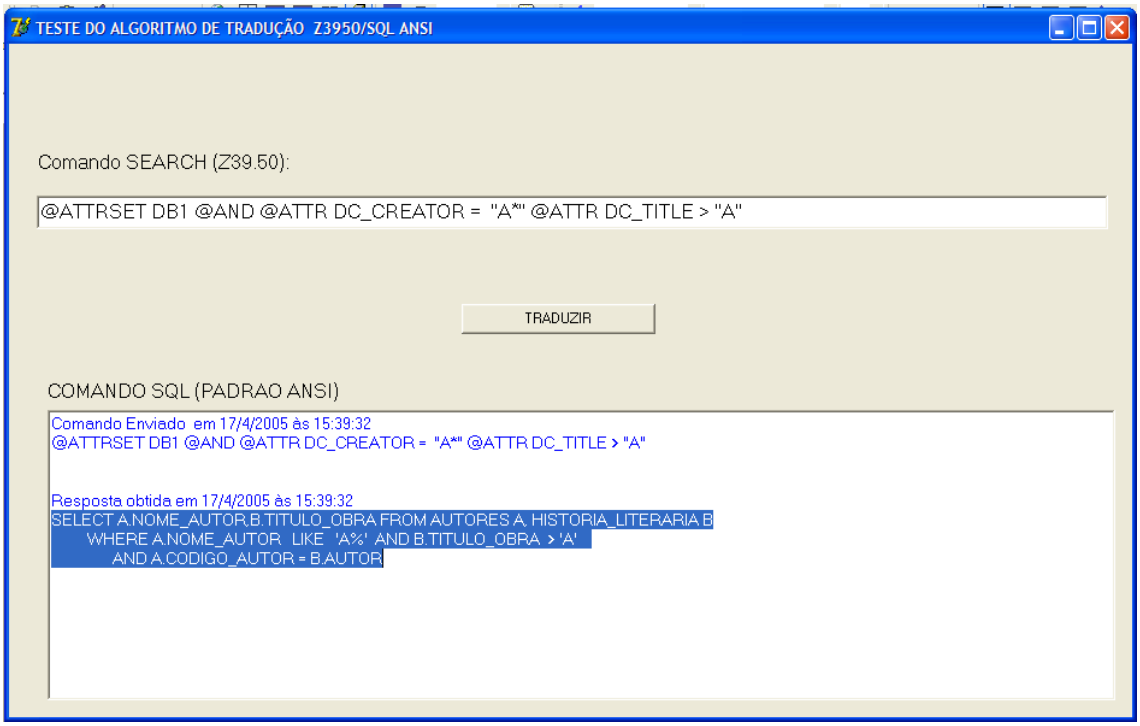


Figura 19. Tradução de um comando Search em SQL e sua execução

A figura 20 mostra um outro teste de tradução de uma query tipo 1, que envolve o relacionamento de dois metatados (Autor e tittulo). Nesta consulta, a intenção foi a de recuperar o titulo das obras de todos os autores cujo nome inicia com a letra “A”.



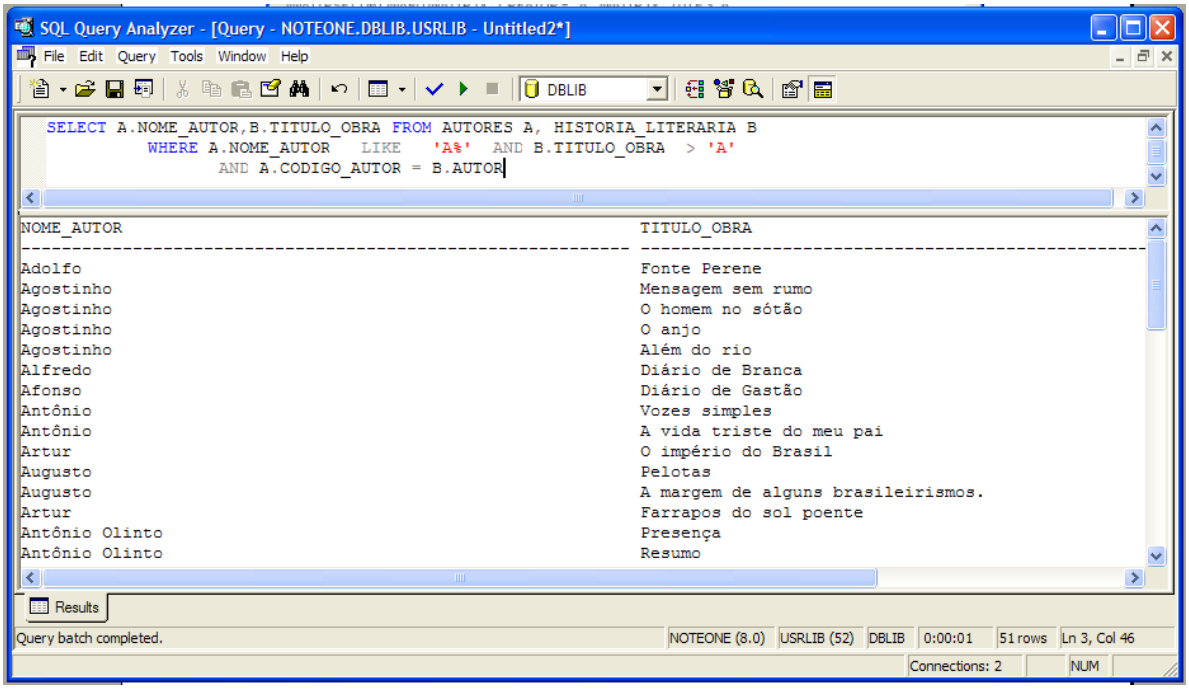


Figura 20. Um comando Search (com dois metadados) e sua tradução em SQL.

6.5 Conclusão

A linguagem DCM-BIB e o modulo de software DCM-Z3950-SQL vão permitir a exposição de um acervo digital proprietário a funcionalidade de busca do Z39.50 com uso dos metados Dublin Core. A solução apresentada reduzirá sensivelmente os esforços requeridos para se alcançar tal exposição, visto que elimina a necessidade de reprogramação e adaptação do esquema do banco de dados. Além de facilitar, através do uso de uma interface gráfica, a atividade de mapeamento dos metadados proprietários em metadados Dublin Core. Desta forma, importantes acervos digitais proprietários serão mais facilmente integrados aos padrões estabelecidos para as bibliotecas digitais. E com isto acreditamos ser esta uma contribuição no sentido de tornar factível o acesso ao acervo das bibliotecas digitais proprietárias.

Capítulo 7. Conclusões

Os avanços técnicos, científicos e culturais de uma sociedade ocorrem principalmente pela utilização do conhecimento já produzido. Este conhecimento tem um valor inestimável, à medida que pode ser colocado a serviço da humanidade para a continuidade de sua evolução.

Cada vez mais, o acervo do conhecimento humano já produzido está sendo digitalizado e armazenado em bibliotecas digitais espalhadas pelo mundo. A interoperabilidade entre elas encontra alguns obstáculos que por sua vez, tornam difícil a pesquisa a um considerável acervo digital.

Uma notável iniciativa foi disponibilizada através da solução Z39.50 (composta de um protocolo e serviços) que permite a busca e recuperação de informações em bibliotecas digitais geograficamente dispersas. Apesar da contribuição do Z39.50, outro obstáculo ainda existe: a falta da adoção de um padrão de metadados que descreve o acervo digital. Felizmente, o padrão de metadados Dublin Core está sendo amplamente utilizado pelas instituições mantenedoras de bibliotecas digitais, tornando possível o estabelecimento da interoperabilidade entre elas.

Existem, contudo, importantes acervos que ainda não aderiram ao padrão Dublin Core e que se encontram armazenados em bibliotecas digitais denominadas aqui de proprietárias. A exposição do acervo das bibliotecas digitais proprietárias as facilidades de busca do Z39.50 encontram suporte em algumas soluções, que contudo, exigem um considerável esforço para adaptação do código do Z39.50 ao esquema do banco de dados que armazena o acervo digital proprietário além de não estabelecer uma correlação entre os metadados proprietários e os metadados Dublin Core dificultando a busca e recuperação de informações.

Neste trabalho propomos uma linguagem, denominada DCM-BIB para definir a correlação entre os metadados Dublin Core e os proprietários. Sua vantagem encontra-

se, na facilidade de uso por parte do administrador da biblioteca digital. A estrutura da linguagem, definida previamente por um esquema XML validado, permite ao administrador produzir um documento XML que descreve as correlações entre os metadados e ainda o relacionamento das tabelas do banco de dados. Para tornar ainda mais fácil e prático o mapeamento dos metadados proprietários em Dublin core, foi criada uma interface onde o administrador informa os dados para produzir um arquivo completo e valido na linguagem DCM-BIB. Para que fosse possível a validação completa da linguagem em questão, criamos um algoritmo que foi implementado dando origem a um protótipo que denominamos de DCM-Z3950-SQL. O protótipo foi submetido a uma seqüência de queries conformantes ao padrão Z39.50 e produziu queries no padrão SQL, obedecendo rigorosamente as correlações definidas no documento DCM-BIB. As queries no padrão SQL foram executadas contra uma base de dados que simulou uma porção de uma biblioteca digital. O resultado dos ensaios realizados foi satisfatório e confirmaram a precisão do algoritmo de conversão de queries, além de provar a inteligibilidade da linguagem DCM-BIB.

A linguagem DCM-BIB e o modulo de software DCM-Z3950-SQL possuem algumas restrições: operam somente em base de dados relacional, trata somente a query tipo 1 (RPN) do padrão Z39.50 e utiliza somente os metadados Dublin Core.

Como implementação futura sugere-se a adaptação de um código aberto do Z39.50 que utilizaria os serviços do modulo DCM-Z3950-SQL para a conversão das queries submetidas a ele e ainda a possibilidade de um mapeamento flexível entre metadados que não sejam os Dublin Core nos metadados proprietários.

Capítulo 8. Referências

(Baker, 1998) Languages for Dublin Core. Disponível em : http://www.dlib.org/dlib/december98/12baker.html Acessado em : Janeiro/2004

(Baker, 2000) A Grammar of Dublin Core. Disponível em : http://www.dlib.org/dlib/october00/baker/10baker.html Acessado em : Janeiro/2004

(BDB, 2005) Brazilian Digital Library. Disponível em http://www.ibict.br. Acessado em Fevereiro de 2005.

(Bordin, 2002) Extensão do Protocolo Z39.50 para Busca de Servidores de Bibliotecas Digitais. Dissertação de Mestrado do curso de Ciência da Computação da UFSC, 2002.

(DCMI, 2001) Using Dublin Core. Disponível em : http://www.dublincore.org/documents/2001/04/12/usageguide/ Acessado em : Janeiro/2004

(DCMI, 2004) Using Dublin Core type. Disponível em : http://dublincore.org/documents/dcmi-type-vocabulary. Acessado em: Julho/2004

(DCMI, 2003) DCMI Glossary. Disponível em: http://www.dublincore.org/documents/usageguide/glossary.shtml. Acessado em: Março/2004

(DCMI, 2005) Dublin Core Metadata Initiative. Disponível em: http://www.dublincore.org/ Acessado em: Março/2004

(DLF, 1998) Digital Library Federation. Disponível em: http://www.diglib.org/about/dldefinition.htm Acessado em: Março/2004

(DLI, 1998) Digital Libraries Initiative. Disponível em: http://www.dli2.nsf.gov./dilione Acessado em: Fevereiro/2005

(Feng,2005) Towards Knowledge-Based Digital Libraries Disponível em: http://citeseer.ist.psu.edu/feng01towards.html. Acessado em: Fevereiro/2005

(Gonçalves, 2001) Flexible Interoperability in a Federated Digital Librray of Theses and Dissertations. CiteSeer, 2001.

(Hammer, 2000) YAZ ToolKit . Disponível em: http://www.indexdata.dk/yaz/. Acessado em: Novembro/2004

(Ibbo, 2000) Pure Java Z3950 ToolKit. Disponivel em http://sourceforge.net/projects/jzjkit/. Acessado em: Março/2004

(Lightle, 2003) Using Metadata Standards to Support Interoperability. Disponível em: http://morty.uts.ohio-state.edu/learning_objects/ Acessado em: Fevereiro/2005

(Miller, 2003) Presserving mapping consistency under schema changes. VLDB , 2003.

(Miller, 2004a) ToMAS: A System for Adapting Mappings while Schemas Envolve. IEEE, 2004.

(Miller, 2004b) Presserving mapping consistency under schema changes. The VLDB Journal, 2004.

(OAI, 2004) Open Archives Initiative Protocol for Metadata Harvesting. Disponível em http://www.openarchives.org/OAI/openarchivesprotocol.html#DefinitionsConcepts. Acessado em: Março/2004

(OAI, 2005) Open Archives Initiative Protocol for Metadata Harvesting. Disponível em http://www.openarchives.org/OAI/openarchivesprotocol.html. Acessado em: Março/2004

(Pistori, 2000) Pistori, Jéferson. Arquitetura de Implementação de uma Biblioteca digital multimídia distribuída. Dissertação de Mestrado do curso de Ciência da Computação da UFSC, 2000.

(Powell, 2002) Dublin Core Management. Disponível em : http://www.ariadne.ac.uk/issue10/dublin/. Acessado em : Janeiro/2003

(Pullian, 1996) Pullian D., Allen J., Clagett J., *Digital Libraries: A Technology Assessment by Benjamin Franklin Scholars*. For Benjamin Franklin Capstone Course (E 467S). North Carolina State University. Disponível em: http://www4.ncsu.edu/unity/users/j/jherkert/dlta.html. 1996. Acessado em: Março 2003

(RDF, 1999) Resource Description FrameWork Disponível em: http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/metadata Acessado em: Março 2003

(SAX, 2005e) SAX Simple API for XML Disponível em: http://www.saxproject.org/ Acessado em: Fevereiro/2005

(Stuer,2005) Internet support for the active use of digital museum data for teaching an presentation. Disponível em : http://portal.acm.org/citation.cfm?id=964455 Acessado em: Fevereiro/2005

(Sumner,2005) Looking at Digital Library Usability from a reuse Perspective Disponível em: http://portal.acm.org/citation.cfm?id=379742 Acesso em Fevereiro/2005

(Taylor, 2002) Generic Z39.50-to-Relational Database gateway. Disponível em http://sql.z3950.org/docs/Net/Z3950/DBIServer.html Acessado em: Fervereiro/2005

(Velagrakis, 2000) On Z39.50 wrapping and description logics. ICS-FORTH, 2000.

(Velagrakis, 2004) Representing and Querying Data Transformations. AT&T Labs-Research , 2004

(Velegrakis, 2005) Declarative Specification of Z39.50 Wrappers Using Description Logics.. Disponível em: http://www.ics.forth.gr/isl/publications/paperlink/dlandz39_50.htm Acessado em: Janeiro/2005

(W3C, 2001a) XML Scheme. Disponível em: http://www.w3.org/XML/Schema Acessado em: Fevereiro/2005

(W3C, 2001b) Validator for XML Schema. Disponível em http://www.w3.org/2001/03/webdata/xsv

(W3C, 2005a) W3C. Extensible Markup Language. Disponível em http://www.w3.org/XML/. Acessado em : Fevereiro de 2005.

(W3C, 2005b) XSL Extensible Stylesheet Language. Disponível em: http://www.w3.org/Style/XSL/ Acessado em: Fevereiro/2005.

(W3C, 2005c) CSS Cascading Style Sheets. Disponível em: http://www.w3.org/Style/CSS/. Acessado em: Fevereiro/2005

(W3C, 2005d) DOM Document Object Model. Disponível em: http://www.w3.org/DOM/. Acessado em: Fevereiro/2005

(W3C, 2005f) XLink XML Linking Language Disponível em: http://www.w3.org/XML/Linking Acessado em: Fevereiro/2005

(W3C, 2005g) XPointer. Disponível em: http://www.w3.org/XML/Linking Acessado em: Fevereiro/2005

(Weibel, 2000) Weibel, Miller, An Introduction to Dublin Core. Disponível em : http://www.xml.com/pub/a/2000/10/25/dublincore/

(Z39.50, 1995) ANSI/NISO. Information Retrieval (Z39.50): Application Service Definition And Protocol Specification. Disponível em http://www.niso.org/standards/resources/Z39-18-1995.pdf Acessado em : Janeiro/2004