Um mecanismo para identificação, representação e consulta de versões de objetos XML oriundos de bibliotecas digitais

Eduardo N. Borges, Renata M. Galante

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil
[eduardo.borges, galante]@inf.ufrgs.br

Abstract. *In distributed information systems it is possible to find multiple representations of the same resource. For instance, a scientific paper indexed by several digital libraries. Aiming to eliminate redundant query results, this work presents a mechanism to detect, represent and query versions of XML objects from different digital libraries. We have defined some similarity functions applied to the digital libraries domain. Also, we have done some experiments for evaluating the performance of the version detection mechanism. Besides eliminating the results redundancy, this works also aims to keep information regarding the retrieved data provenance.*

Resumo. *Em sistemas de informação distribuídos é possível existir múltiplas representações de um mesmo recurso. Um artigo científico indexado por diversas bibliotecas digitais é um exemplo desse recurso. Visando eliminar resultados redundantes das consultas, este trabalho apresenta um mecanismo para detectar, representar, e consultar versões de objetos XML provenientes de diferentes bibliotecas digitais. Foram definidas funções de similaridade aplicadas ao domínio das bibliotecas digitais. Também foram realizados experimentos responsáveis por avaliar a eficiência do mecanismo de detecção de versões. Além de eliminar a redundância dos resultados, este trabalho também visa manter informações a respeito da proveniência dos dados recuperados.*

1. Introdução

Em sistemas de informação distribuídos é possível existir múltiplas representações de um mesmo recurso. A replicação de recursos ou objetos na rede pode ser uma grande vantagem para acelerar as consultas sobre ambientes distribuídos. Neste contexto, é necessário um processo de identificação ou detecção destas múltiplas representações do mesmo objeto compartilhado.

Exemplos de objetos compartilhados são artigos científicos indexados por diversas bibliotecas digitais. As bibliotecas digitais são caracterizadas por repositórios de dados que possuem meta-informações sobre os artigos científicos publicados e apontadores para edições digitais destes artigos. Estas edições geralmente encontram-se nos formatos *Post Script* ou PDF. As bibliotecas digitais representam informações de fontes heterogêneas, porém possuem certa estrutura, mais regular que a Web e menos

regular que os bancos de dados. Além disso, possuem seu conteúdo mais controlado em vista de possuírem um público alvo. Um exemplo de biblioteca digital é a Biblioteca Digital Brasileira de Computação (BDBComp) [Laender, Gonçalves e Roberto 2004] que constitui-se em um repositório de artigos científicos em Computação publicados no Brasil.

Atualmente, muitos usuários realizam buscas na Web sobre informações de artigos científicos, os quais estão dispostos em várias bibliotecas digitais como *Digital Bibliography & Library Project* (DBLP) [University of Trier 2007], *ACM Digital Library* [Association for Computing Machinery 2007], *IEEE Computer Society Digital Library* [Institute of Electrical and Electronics Engineers 2007], entre outras. Um mesmo artigo científico pode ser referenciado por várias bibliotecas digitais, mas a representação desta referência é diferente para cada sistema. O nome dos autores de determinado artigo, por exemplo, pode estar armazenado de diversas formas. Os metadados associados ao autor do artigo são diferentes em cada biblioteca digital.

O formato XML [World Wide Web Consortium 2007] vem sendo adotado como principal forma de representação de metadados associados a sistemas de informação em geral, incluindo bibliotecas digitais. Para o suporte a integração e consulta em diversas bibliotecas digitais é necessário o casamento dos metadados XML correspondentes em cada biblioteca digital, assim como identificar diferentes representações das mesmas instâncias. Além disso, deve-se representar e consultar versões destes metadados.

Dentro deste contexto, o objetivo deste trabalho de mestrado é especificar um mecanismo para detectar, representar e consultar versões de objetos XML[1] provenientes de diferentes bibliotecas digitais. A abordagem proposta visa representar as versões sem perder informações sobre a proveniência desses objetos, de forma que seja possível realizar consultas sem perder as informações relativas à origem dos atributos dos objetos XML. Ao mesmo tempo, o resultado retornado ao usuário deve ser o mais completo possível. No primeiro caso, o mecanismo de versões é capaz de identificar a fonte da informação dos atributos dos objetos XML através de um modelo de proveniência de dados. No segundo caso, o mecanismo gera para o usuário o máximo possível de informação integrada das bases de dados referentes ao artigo científico consultado.

A principal contribuição deste trabalho é fornecer ao usuário uma resposta livre de redundância para consultas a objetos XML obtidos através da Web, sem perder informações da origem dos dados. A principal aplicação da abordagem proposta é na área de bibliotecas digitais, na consulta a metadados de artigos científicos. Outra aplicação que se beneficiaria da proposta é na área de sistemas distribuídos, na consulta por versões e réplicas de documentos XML em redes P2P.

O restante do texto está organizado da seguinte forma. Na seção 2 é apresentado o contexto no qual a dissertação está inserida, fazendo uma curta revisão bibliográfica sobre proveniência de dados e técnicas utilizadas para o cálculo da similaridade entre documentos XML. A seção 3 define o mecanismo proposto para detecção, representação e consulta a objetos XML oriundos de bibliotecas digitais, definindo a arquitetura do sistema e identificando as principais etapas do trabalho proposto. Na seção 4 são

[1] O termo *objeto XML* utilizado neste artigo refere-se a quaisquer informações representadas no formato XML, as quais podem estar contidas tanto em parte quanto em mais de um arquivo XML.

definidas duas funções de similaridade específicas para o domínio das bibliotecas digitais. Na seção 5 são abordadas as possíveis áreas de pesquisa futura na especificação de um modelo de proveniência de dados para o mecanismo de detecção proposto. Experimentos já realizados e a validação proposta são demonstrados na seção 5. Por fim, na seção 6, são expostas as considerações parciais e próximas atividades.

2. Base Conceitual

Esta seção apresenta o contexto no qual a dissertação está inserida. As áreas de pesquisa relacionadas são brevemente apresentadas, assim como seus inter-relacionamentos, visando delimitar o escopo do problema a ser tratado. São levantadas algumas técnicas utilizadas para o cálculo da similaridade entre documentos XML baseadas na estrutura e no conteúdo. Além disso, são abordados conceitos sobre proveniência de dados e suas principais características.

2.1. Funções de similaridade aplicadas a XML

As técnicas de similaridade entre documentos XML geralmente utilizam algoritmos de detecção de diferenças entre documentos XML, também conhecidos como algoritmos *diff*. Grande parte destes algoritmos retorna um documento ou *script* (*delta*) contendo as operações básicas necessárias para transformar um documento em outro. O conjunto de diferenças fornecido como resposta destes algoritmos não é suficiente para a análise semântica do grau de similaridade entre os documentos analisados. É necessário um coeficiente de similaridade entre os documentos XML que possa classificá-los como versões ou instâncias diferentes.

Algumas propostas tradicionais consideram somente o conteúdo textual dos documentos [Baeza-Yates & Ribeiro-Neto 1999]. Toda informação estrutural é descartada. Somente o conteúdo dos elementos XML é utilizado na comparação. Outras abordagens mais recentes analisam também a estrutura dos documentos [Dorneles et al 2004] [Lian et al 2004] [Joshi et al 2003] [Flesca et al 2002] [Flesca et al 2005] [Nierman & Jagadish 2002].

Uma das aplicações da similaridade entre documentos XML é o agrupamento (*clustering*) de documentos XML. Esta aplicação consiste em agrupar documentos similares em estrutura e/ou conteúdo visando aumentar o desempenho de consultas realizadas sobre estes conjuntos de documentos. Um método visando o agrupamento de documentos XML, baseado na similaridade estrutural, é proposto por Nierman e Jagadish [2002]. Esta abordagem modela os documentos XML em árvores rotuladas ordenadas e utiliza uma modificação do algoritmo *tree edit distance* [Chawathe & Garcia-Molina 1997] para medir a similaridade estrutural entre documentos.

Os algoritmos de similaridade entre documentos XML possuem alto custo computacional. Quando aplicados a coleções muito grandes podem tornar-se inviáveis. Para contornar este problema, algumas técnicas foram propostas a fim de diminuir o tempo de processamento necessário para o cálculo da similaridade. Em [Lian et al 2004] é proposto um algoritmo denominado S-GRACE para agrupamento de documentos XML baseado na estrutura dos dados. A noção de grafo de estruturas (*s-graph*) é proposta, suportando uma métrica de distância entre documentos e coleções de

documentos mais eficientes, ou seja, com um custo computacional baixo quando comparada a outras métricas baseadas em *tree-edit distance*.

Os documentos XML possuem a característica de conterem dados semi-estruturados. Por esse motivo, dados representados pelo conteúdo de um documento podem estar presentes na estrutura de outro. Isto dificulta ainda mais a análise da similaridade entre documentos XML. Apesar de várias abordagens terem sido propostas, a análise semântica do grau de similaridade entre dois documentos XML – de modo a classificá-los como versões de um mesmo documento ou instâncias diferentes ainda é um problema em aberto pela comunidade científica.

Dentro desse contexto, um dos objetivos específicos deste trabalho de mestrado é estudar o problema da similaridade de documentos XML, considerando o conteúdo e a estrutura, de forma a classificar metadados no formato XML, os quais descrevem artigos científicos, como versões de um mesmo artigo científico ou instâncias diferentes.

2.2. Proveniência de dados

Com o grande número de conjuntos de dados aparecendo em domínio público, torna-se cada vez mais necessário determinar a veracidade e qualidade destes dados. Um histórico detalhado dos dados permite aos usuários avaliar se estes dados são aceitáveis e confiáveis. Uma nova linha de pesquisa denominada *data provenance* [Buneman, Khanna e Tan 2000] tem proposto soluções para este problema.

A proveniência de dados vem sendo descrita de várias maneiras dependendo do domínio em que ela é aplicada. Conhecida também como *data pedigree* ou *data lineage*, a proveniência de dados é a descrição das origens de uma porção de dados e o processo pelo qual ela é obtida [Buneman, Khanna e Tan 2001]. Lanter [Lanter 1991] caracteriza a proveniência como informações que descrevem as transformações aplicadas para derivar os dados. Greenwood [et al 2003] expande a definição de Lanter dizendo que a proveniência de dados é caracterizada por metadados que descrevem os processos de *workflows* e anotações sobre experimentos. Já Simmhan e Gannon [2005] a definem como informações que ajudam a determinar o histórico de derivação de um produto de dados[2], começando pelas suas fontes de origem.

Duas principais características compõem a proveniência de dados: os produtos de dados ancestrais dos quais os dados foram evoluídos e o processo de transformação, o qual é responsável pela derivação destes produtos de dados. Buneman, Khanna e Tan [2001] definem estas duas características como *where-provenance* – de onde os dados são obtidos, ou seja, a origem de uma porção de dados – e *why-provenance* – por que esta porção de dados está em um determinado banco de dados. A proveniência pode ser dividida de acordo com outras características como:

- Aplicação: sistemas de proveniência podem suportar diversos tipos de uso. Podem ser usados para estimar a qualidade dos dados, visando o reuso das informações. Informações de proveniência também podem ser usadas para determinar a autoria dos dados, entre outras atribuições.

[2] O termo *produto de dados* refere-se a dados em qualquer forma de representação, por exemplo: tabelas, arquivos, coleções, etc.

- Orientação: a proveniência pode ser orientada a dados ou orientada a processos (transformações dos dados).
- Granularidade: é flexível de acordo com o contexto. Por exemplo, em um banco de dados, a granularidade da proveniência pode estar definida para atributos de uma relação, ou para uma relação inteira.
- Representação: as principais formas de representar a proveniência são anotações e inversões. As anotações são metadados que expressam o histórico da derivação dos dados, sua origem e os processos envolvidos na derivação. Quando as derivações podem ser invertidas a fim de recuperar os dados de origem, são utilizadas inversões para representar a proveniência. Apesar das inversões serem mais compactas, elas são limitadas ao histórico de derivação dos dados. Já as anotações incluem informações adicionais, o que enriquece semanticamente a proveniência.
- Armazenamento: informações de proveniência podem ser maiores que os dados descritos por elas. A granularidade da proveniência é inversamente proporcional ao tamanho ocupado pelo armazenamento destas informações. A forma de armazenamento escolhida em um sistema é importante para a sua escalabilidade. Geralmente, inversões são mais escaláveis que anotações. Uma solução é utilizar anotações simples e buscar o histórico de derivação dos dados recursivamente.
- Disseminação: os sistemas de proveniência devem disponibilizar aos usuários métodos de consulta a estas informações. A maneira usual de disseminação se dá através de um grafo de derivação. Outras formas incluem realizar consultas (*queries*) diretamente sobre as informações de proveniência e o uso de APIs de recuperação de proveniência.

Uma taxonomia das técnicas de proveniência definida em [Simmhan & Gannon 2005] é apresentada na Figura 1.

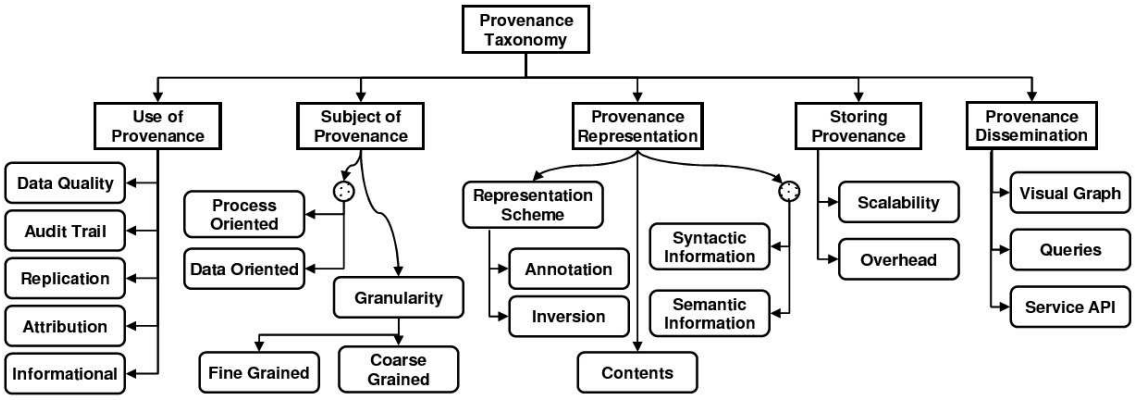

Figura 1. Taxonomia da proveniência de dados [Simmhan & Gannon 2005].

3. Uma visão geral do mecanismo para detecção e representação de versões

Esta seção apresenta uma visão geral da solução proposta para o mecanismo de detectar, representar e consultar versões de objetos XML provenientes de diferentes bibliotecas digitais. A solução proposta apresenta seu diferencial ao fornecer ao usuário

que consulta objetos XML obtidos da *Web* uma resposta livre de redundância e, ao mesmo tempo, mantendo informações de proveniência dos dados recuperados.

Considere um ambiente de integração de bibliotecas digitais formado pela BDBComp e DBLP, por exemplo. Um usuário submete uma consulta por nomes de autores: "Edleno Silva de Moura, Altigran Soares da Silva". Cada biblioteca digital armazena estas informações de maneira diferente. Os metadados que descrevem os autores associados ao artigo científico são heterogêneos (Figura 2). O elemento `criator` presente nos metadados da BDBComp (linhas 03-07) corresponde ao elemento `author` na DBLP (linhas 19-23). As representações, apesar de diferentes, fazem referência à mesma informação.

BDBComp

```
01 <oaidc:dc> → atributos omitidos
02    <title>Detec&#231;&#227;o de Sitios Replicados Utilizando Conte&#250;do e Estrutura</title>
03    <creator>Andr&#233; Luiz da Costa Carvalho</creator>
04    <creator>Allan Jos&#233; de Souza Bezerra</creator>
05    <creator>Edleno Silva de Moura</creator>
06    <creator>Altigran Soares da Silva</creator>
07    <creator>Patr&#237;cia Silva Peres</creator>
08    <description>Identifying replicated sites is an important task for search engines.It can reduce
data storage costs, improve query processing time and remove noises that might affect the quality of
the final answer given to the user . This paper introduces a new approach to detect replicated sites in
search engines databases, using as replication evidences the websites&apos; structure and the content
of their pages. It is also depicted the result of experiments performed with a real search engine
database. Our approach found 8.43% of the web pages stored in the database were in replicated web
sites with 94.4% precision, result witch is more accurate than the ones found in other
works.</description>
09    <date>2005</date>
10    <type>Text</type>
11    <identifier>sbbd2005article002</identifier>
12    <identifier>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-02-novo_Carvalho.pdf</identifier>
13    <source>sbbd2005</source>
14    <language>por</language>
15    <coverage>Uberl&#226;ndia, MG, Brasil</coverage>
16    <rights>Sociedade Brasileira de Computa&#231;&#227;o</rights>
17 </oaidc:dc>
```

DBLP

```
18 <inproceedings> → atributos omitidos
19    <author>Andr&amp;eacute; Luiz da Costa Carvalho</author>
20    <author>Allan Jos&amp;eacute; de Souza Bezerra</author>
21    <author>Edleno Silva de Moura</author>
22    <author>Altigran Soares da Silva</author>
23    <author>Patr&amp;iacute;cia Silva Peres</author>
24    <title>Detec&amp;ccedil;&amp;atilde;o de R&amp;eacute;plicas Utilizando
Conte&amp;uacute;do e Estrutura.</title>
25    <pages>25-39</pages>
26    <year>2005</year>
27    <crossref>conf/sbbd/2005</crossref>
28    <booktitle>SBBD</booktitle>
29    <ee>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-02-novo_Carvalho.pdf</ee>
30    <url>db/conf/sbbd/sbbd2005.html#CarvalhoBMSP05</url>
31 </inproceedings>
```

Figura 2. Metadados heterogêneos associados ao mesmo artigo científico.

BDBComp + DBLP

```
01 <metadata>
02    <title>Detecção de Réplicas Utilizando Conteúdo e Estrutura</title>
03    <author>André Luiz da Costa Carvalho</author>
04    <author>Allan José de Souza Bezerra</author>
05    <author>Edleno Silva de Moura</author>
06    <author>Altigran Soares da Silva</author>
07    <author>Patrícia Silva Peres</author>
08    <abstract>Identifying replicated sites is an important task for search engines.It can reduce data
storage costs, improve query processing time and remove noises that might affect the quality of the
final answer given to the user . This paper introduces a new approach to detect replicated sites in
search engines databases, using as replication evidences the websites' structure and the content of
their pages. It is also depicted the result of experiments performed with a real search engine database.
Our approach found 8.43% of the web pages stored in the database were in replicated web sites with
94.4% precision, result witch is more accurate than the ones found in other works.</abstract>
09    <year>2005</year>
10    <fulltext>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-02-novo_Carvalho.pdf</fulltext>
11    <language>Portuguese</language>
12    <booktitle>SBBD</booktitle>
13    <pages>25-39</pages>
14    <city>Uberlândia, MG, Brasil</city>
15    <rights>Sociedade Brasileira de Computação</rights>
16 </metadata>
```

Figura 3. Possível retorno de uma consulta, com informações combinadas de ambas as versões de metadados.

Além da heterogeneidade dos metadados, a codificação de caracteres pode ser diferente para cada biblioteca (linhas 04 e 20). Quanto ao conteúdo dos metadados, outro problema é identificado. Diferentes representações podem ser encontradas para um mesmo artigo científico. O metadado `title` assume o valor "Detecção de `Sítios Replicados` Utilizando Conteúdo e Estrutura" na BDBComp, enquanto "Detecção de `Réplicas` Utilizando Conteúdo e Estrutura" na DBLP (linhas 02 e 24).

Portanto, é necessário repassar ao usuário uma resposta única, com o máximo possível de informações sobre os artigos científicos recuperados na consulta, combinadas e extraídas das bibliotecas digitais. Além disso, esta resposta deve ser livre de redundância (Figura 3).

Do ponto de vista do usuário, é importante que seja retornado o resultado com todos os dados possíveis que descrevem o artigo científico (cada parte da informação recuperada de uma determinada biblioteca digital). Mas, para o sistema é necessária uma estrutura para representar as versões dos metadados e o local de origem destas informações. Modelos de proveniência de dados podem ser especificados no intuito de rastrear e armazenar as origens destas informações e a movimentação das mesmas entre as diversas bibliotecas digitais [Simmhan & Gannon 2005].

No intuito de identificar as diferentes representações de uma mesma publicação em sistemas de integração de bibliotecas digitais, torna-se necessário um mecanismo automático para detecção e representação de versões. Este mecanismo possui como objetivo descobrir se metadados no formato XML (provenientes de diversas fontes de integração) representam versões do mesmo objeto (artigo científico) ou instâncias

diferentes. O mecanismo deve fornecer uma estrutura de dados adequada para o armazenamento e representação das versões de objetos XML, fornecendo ao usuário respostas livres de redundância, abstraindo o conceito de versões. Cabe ressaltar que o trabalho tem como escopo metadados relativos a artigos científicos.

A abordagem utilizada para cumprir o objetivo está dividida em quatro grandes etapas. A Figura 4 mostra a arquitetura do mecanismo onde existe um módulo do sistema para cada uma das três primeiras fases do trabalho. Primeiramente, pretende-se adaptar técnicas de similaridade entre documentos XML, considerando o conteúdo e a estrutura, a fim de detectar versões dos objetos XML referentes aos artigos científicos. O módulo de `identificação` é responsável por esta tarefa.

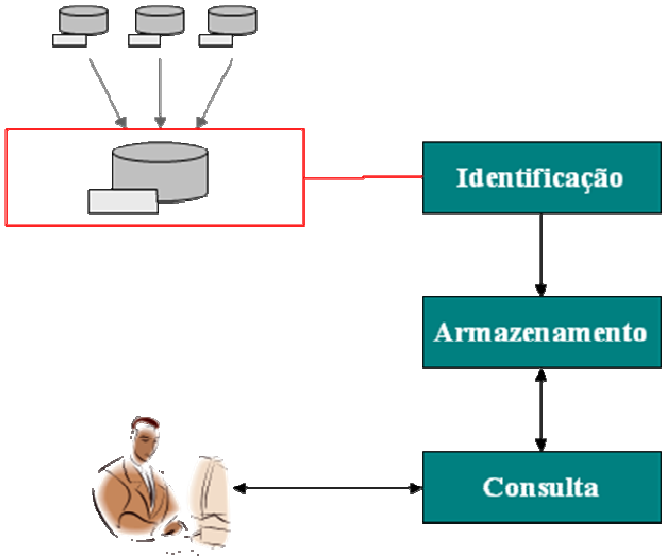

Figura 4. Arquitetura do sistema proposto.

Com os objetos XML identificados, a segunda etapa consiste em especificar um modelo de versões para o `armazenamento` das versões dos objetos XML. É prevista a especificação de um modelo de proveniência de dados, responsável por rastrear as informações de proveniência dos atributos dos objetos XML.

Na terceira etapa, pretende-se dar suporte a consultas de forma a permitir realizar buscas pela proveniência dos dados. Através do módulo de `consulta`, o usuário final poderá realizar consultas sem ter conhecimento sobre a existência de versões, obtendo uma resposta única (versões integradas, ou seja, a totalidade de informação relevante contida nas versões) para os artigos científicos pesquisados. Além disso, serão recuperadas informações de proveniência dos dados obtidos na resposta à consulta realizada.

Por fim, durante a quarta etapa, o mecanismo proposto será validado através de dois tipos de experimentos. O primeiro conjunto de experimentos é responsável por avaliar o desempenho e eficiência da detecção de versões dos objetos XML. O segundo conjunto deve comparar a abordagem proposta com outros trabalhos na área que não consideram versões e proveniência dos dados.

Essa seção apresentou uma visão geral da dissertação com o intuito de delinear objetivos propostos, através de exemplos esclarecedores que ilustrassem o problema e

cada uma das etapas propostas para a solução. Encerrada essa visão geral, as próximas seções especificam em detalhes os objetivos já alcançados e as próximas atividades a serem realizadas.

4. Identificação das versões

A Figura 5 ilustra o módulo de identificação de versões de objetos XML apresentado na arquitetura do sistema. Os arquivos XML são referenciados por um número composto do identificador do documento, seguido do caractere "." (ponto), seguido do identificador da versão. Réplicas ou duplicatas estão representadas pelo mesmo número. Para a implementação do mecanismo de detecção é necessário mapear os diferentes metadados provenientes de cada biblioteca digital. O casamento entre esquemas XML é amplamente estudado em trabalhos como [Rahm & Bernstein 2001] e neste trabalho é considerado como um problema já resolvido. Após a fase de casamento, é necessário comparar cada par de referências disponíveis nos arquivos XML no intuito de identificar as diferentes representações de um mesmo documento (artigo científico publicado).

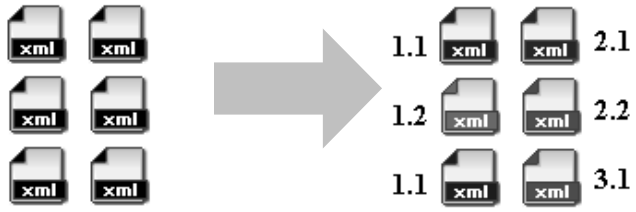

Figura 5. Módulo de identificação de versões de objetos XML.

Visando o cálculo da similaridade em tempo computacional não proibitivo, foram propostas abordagens que não utilizam o modelo tradicional de árvores para a representação dos documentos XML. Os resultados em relação à qualidade da saída gerada por estas abordagens são comparáveis aos algoritmos de edição de árvores. Além disso, estas métricas possuem tempo de execução muito menor e são classificados como algoritmos rápidos. Com o grande crescimento das bases de dados XML, como é o caso das bibliotecas digitais, ainda é necessário o desenvolvimento de técnicas que possam ser processadas em tempo computacional hábil, com complexidade linear, visando a escalabilidade. Visto que nenhuma solução ótima linear de propósito geral foi desenvolvida, algoritmos de similaridade entre documentos XML específicos para certos domínios devem ser utilizados.

Algumas bibliotecas digitais possuem um número de referências muito alto. A DBLP, por exemplo, conta com mais de 800 mil artigos catalogados. Comparar cada um destes artigos com todos os artigos de outras bibliotecas digitais é uma tarefa que exige um custo computacional altíssimo. Para que o número de comparações entre as instâncias de cada biblioteca digital possa ser reduzido, foram propostas duas métricas de similaridade entre os autores dos artigos, denominadas *nomesIni* e *simNomes*.

Considere um conjunto $I_S = \{x \in I \mid x = 0 \vee x = 1\}$ como o conjunto dos valores inteiros em I_S no intervalo [0,1], e um conjunto $P = \{y \mid y = [A\text{-}Z]^*\}$ como o conjunto de todas as palavras formadas por quaisquer caracteres alfabéticos. A função de

similaridade $nomesIni : P \rightarrow I_S$ recebe um par de palavras $a, b \in P$ e gera um valor de similaridade $s \in I_S$. A Figura 6 define a função de similaridade *nomesIni*.

$$nomesIni\ (a, b) = \begin{cases} 1, & (a_1 = b_1 \wedge a_m = b_n) \vee \\ & (a_1 = b_2 \wedge a_m = b_1) \vee \\ & (a_1 = b_1 \wedge a_2 = b_2) \vee \\ & (a_1 = b_n \wedge a_2 = b_1) \\ 0, & \text{caso contrário} \end{cases}$$

Figura 6. Definição da função *nomesIni*.

Onde a_i é a i-ésima letra da palavra *a*, b_i é i-ésima letra da palavra *b*, *m* é o tamanho da palavra *a*, e *n* é o tamanho da palavra *b*. A função de similaridade *nomesIni* recebe como parâmetro duas seqüências de caracteres *(a, b)* que representam as iniciais de dois autores que publicaram artigos em uma biblioteca digital. Quando as iniciais correspondem ao mesmo autor, ou seja, quando as duas representações podem expressar o mesmo objeto do mundo real, a função *nomesIni* retorna um valor de similaridade $s = 1$ (um). Caso contrário, é retornado o valor $s = 0$ (zero). As condições impostas pela função para o casamento das iniciais partem do princípio que o nome de um autor pode estar representado de diversas maneiras como mostra o exemplo na Tabela 1.

Tabela 1. Possíveis representações de "Eduardo Nunes Borges".

Nome	Iniciais
Eduardo N. Borges	ENB
E. Borjes	EB
Borges, Edward	BE
Borjes, E. Nuñes	BEN

Considere um conjunto $R_S = \{x \in R \mid x \geq 0 \wedge x \leq 1\}$ como o conjunto de todos os valores reais em R_S no intervalo [0,1]. A função de similaridade $simNomes : \{R_S, P\} \rightarrow I_S$ recebe duas listas de palavras *K* e *L* – onde cada elemento das listas $K_i, L_i \in P$ – e um valor de limiar (*threshold*) de similaridade $t \in R_S$, e gera um valor de similaridade $s \in I_S$. A Figura 7 define a função de similaridade *simNomes*.

$$simNomes\ (K, L, t) = \begin{cases} 1, & \dfrac{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(nomesIni\left(K_i, L_j\right)\right)}{\max(m,n)} \geq t \\ 0, & \text{caso contrário} \end{cases}$$

Figura 7. Definição da função *simNomes*.

Onde *i* é a i-ésima palavra da lista *K*, *j* é a j-ésima palavra da lista *L*, *m* é o tamanho da lista *K*, e *n* é o tamanho da lista *L*. *max(i, j)* é uma função que retorna o tamanho da maior lista de palavras. A função de similaridade *simNomes* recebe como parâmetro duas listas de iniciais de autores que publicaram artigos em uma biblioteca digital. Além destes, um parâmetro real corresponde a um valor de limiar mínimo de casamento entre as iniciais. Este limiar tem como objetivo deixar a função flexível a ajustes e varia no intervalo fechado [0,1]. O casamento entre as palavras é realizado através da função *nomesIni* definida anteriormente. Quando o limiar de casamento mínimo é atingido, a função retorna *s = 1* (um), caso contrário retorna *s = 0* (zero).

Por exemplo, *simNomes((ENB, NI, OCM), (IFN, BE, CO), 1.0)* retorna *1* pois *ENB* corresponde a *BE*, *NI* corresponde a *IFN* e *OCM* corresponde a *CO*. Foi atendido o limiar de 100% de casamentos. Já *simNomes((ENB, NI, OCM), (IFN, BE, CYQ), 0.75)* retorna *0* pois *ENB* corresponde a *BE*, *NI* corresponde a *IFN*, *OCM* e *CYQ* não possuem correspondência. Portando somente 66,6% das ocorrências casaram, não atingindo o limiar mínimo de 75% passado como parâmetro. O limiar de similaridade de 75% adotado neste caso faz com que um artigo possível candidato ao casamento fique de fora da comparação do restante dos metadados. Definições adequadas de valores de limiar são discutidas na literatura [Stasiu, Heuser e Silva 2005] e não fazem parte do escopo do trabalho apresentado neste artigo. Além disso, outros mecanismos como algoritmos de classificação podem ser utilizados para realizar o casamento das instâncias evitando o uso de *threshold* [Saccol, Nina e Galante 2007].

A função de similaridade proposta tende a obter a revocação (*recall*) máxima, ou seja, entre os casamentos de artigos recuperados estarão todos os casamentos relevantes. A precisão (*precision*) é mínima, visto que o número de casamentos recuperados é muito maior que o número de casamentos relevantes. A precisão atinge valores aceitáveis quando as instâncias que satisfizerem as condições impostas por esta métrica de similaridade são avaliadas na totalidade. Esta avaliação total pode ser realizada com qualquer outra função de similaridade. Os outros metadados como título, conferência e data só serão comparados para os pares de artigos em que a função *simNomes* não retorne zero. Isto reduz drasticamente o tempo de processamento no processo de identificação de versões de metadados.

Segundo os experimentos realizados e descritos na seção 5.1, o número de comparações necessárias após a aplicação desta função de similaridade é 0,024%. Além disso, o tempo computacional necessário para aplicar esta função sobre a totalidade das instâncias (iniciais dos nomes dos autores) é muito menor que aplicar algoritmos de similaridade convencionais que geralmente possuem complexidade quadrática em função do tamanho das entradas (nomes dos autores).

4. Armazenamento e proveniência

Baseado nas características da proveniência apresentadas na seção 2.2, pretende-se especificar um modelo de proveniência de dados para o módulo de armazenamento de versões de objetos XML. Este modelo deve levar em conta dados no formato XML oriundos de biblioteca digitais. Informações de proveniência destes dados devem ser armazenadas a fim de identificar as origens dos dados e o processo pelo qual estes dados foram submetidos e assim derivados.

A proveniência do sistema é orientada a dados, com granularidade fina, ou seja, o produto de dados em questão é um metadado qualquer que descreve um artigo, como por exemplo, *title*. A principal aplicação da proveniência é na identificação da autoria[3] dos metadados que descrevem um artigo científico. O modelo de proveniência previsto deve ser "escalável", no sentido de ser expansível a diversas bibliotecas digitais.

Para representar a proveniência dos dados, pretende-se utilizar anotações, pois são semanticamente ricas. Ainda é necessário um estudo mais aprofundado quanto ao armazenamento da proveniência. Já a disseminação será realizada através de consultas. Ainda deve-se considerar se a proveniência é imutável – onde a atualização das fontes de origem dos dados não implica em atualização da proveniência – ou se deve ser atualizada para refletir o estado atual de seus predecessores.

5. Experimentos

O trabalho proposto será validado através de dois tipos de experimentos. O primeiro conjunto de experimentos é responsável por avaliar o desempenho e a eficiência da detecção de versões dos objetos XML. O segundo conjunto deve comparar a abordagem proposta com outros trabalhos na área que não consideram versões e proveniência dos dados.

5.1 Avaliação da detecção de versões.

Dentro do primeiro conjunto foi realizado um experimento preliminar sobre duas bibliotecas digitais – BDBComp e DBLP. O experimento foi realizado em um computador com processador AMD Athlon 2700+, com 1GB de memória RAM, HD SATA 200GB 7200RPM, rodando o sistema operacional Linux Slackware 10.2 e o SGBD PostgreSQL 8.2.

O principal objetivo deste experimento é mensurar um tempo mínimo (utilizando um computador pessoal desktop) necessário para a identificação das diferentes versões de um mesmo artigo científico nestas duas bibliotecas digitais, as quais podem ser caracterizadas pelos seguintes fatores:

- número de referências (artigos científicos publicados);
- disponibilização dos metadados;
- tamanho dos arquivos XML (expresso em MB).

A BDBComp possui cerca de 4 mil referências para artigos científicos publicados no Brasil. Os metadados que descrevem os artigos científicos são disponibilizados através do protocolo OAI-PMH [Van de Sompel et al 2004], no padrão Dublin Core. Foi realizada uma colheita dos metadados (*metadata harvesting*) os quais foram agrupados em um único arquivo XML de 3,63 MB denominado `bdbcomp.xml`. Já a DBLP conta com mais de 800 mil referências para artigos científicos publicados em diversos países. Os metadados são disponibilizados no *web site* principal da biblioteca digital em um único arquivo XML de 350 MB denominado `dblp.xml`. Também é fornecida uma DTD contendo um simples esquema.

[3] O termo *autoria* refere-se a qual biblioteca digital um produto de dados pertence, e não aos autores de um determinado artigo.

Utilizar algum algoritmo de detecção de diferenças com o objetivo de analisar o resultado em busca da similaridade de algumas instâncias não é apropriado. Os algoritmos *diff* são aplicáveis em casos onde as diferenças entre os arquivos XML são menores que as semelhanças.

As técnicas de similaridade apresentadas na seção 2.2 possuem alto custo, tanto de processamento, quanto de memória. Carregar inteiramente os arquivos `bdbcomp.xml` e `dblp.xml` em memória, a fim de compará-los utilizando algoritmos baseados em árvore, é uma tarefa árdua que exige hardware específico e alto poder computacional.

Visando diminuir este problema, foi realizado um pré-processamento dos arquivos XML com o objetivo de carregá-los em um banco de dados relacional. Para este experimento foi desenvolvido um software em JAVA que realiza o *parsing* dos arquivos XML e obtém como saída um *script* SQL gerado para popular uma base de dados do PostgreSQL. A base de teste é composta por duas tabelas, uma para cada biblioteca digital. O esquema da relação `bdbcomp` é especificado na figura 8. A tabela `dblp` foi criada de maneira análoga.

```
01 CREATE TABLE bdbcomp (
02   id bigint NOT NULL,
03   titulo text,
04   autores text,
05   iniciais text,
06   CONSTRAINT bdbcomp_pkey PRIMARY KEY (id)
07 );
08 CREATE INDEX bdbcomp_iniciaisindex ON bdbcomp
09   USING btree (iniciais)
10 ;
```

Figura 8. DDL da Tabela bdbcomp.

Onde `id` (linha 02) corresponde ao identificador único de cada artigo. Este campo é gerado através de um contador no *parser* do software JAVA, o qual é incrementado a cada elemento `oaidc:dc` (Figura 2, linha 01) encontrado em `bdbcomp.xml` ou `inproceedings` (Figura 2, linha 18) em `dblp.xml`. O atributo `titulo` (linha 03) corresponde ao metadado `title` (Figura 2, linhas 2 e 24) para ambas as bibliotecas digitais. Uma lista composta pelos elementos `creator` (Figura 2, linhas 03-07) ou `author` (Figura 2, linhas 19-23) corresponde ao campo `autores` (linha 04) das tabelas. Por fim, o atributo `iniciais` (linha 05) é formado pelas iniciais dos autores separadas por “ ” (espaço). O restante dos metadados foi ignorado por se tratar de um experimento preliminar, onde o principal objetivo é mensurar o um tempo mínimo necessário para a identificação das diferentes versões de um mesmo artigo científico nestas duas bibliotecas digitais.

Foram implementadas duas funções de similaridade, as quais foram anexadas ao SGBD. A função *simNomes*, apresentada na seção 3.1, e a função *levenshtein*[4] [Levenshtein 1966]. Estas funções foram projetadas com o objetivo de possibilitar ao usuário a realização de consultas utilizando-as de forma transparente, pois estas são

[4] A função de similaridade `levenshtein` recebe como parâmetros duas seqüências de caracteres e retorna um valor real de similaridade no intervalo fechado [0,1].

executadas internamente no SGBD. Após a análise realizada pelo `Analisador Léxico e Sintático` do PostgreSQL, o `Sistema de Reescrita` analisa semanticamente a consulta. Nesta etapa, o SGBD traduz as chamadas das funções e reescreve a árvore de análise. Após, são verificadas as regras no catálogo do sistema e é elaborado um plano de execução otimizado. Os dados são buscados no banco através do `Processador de Consultas` e exibidos ao usuário através da interface de conexão.

Na primeira vez que uma função definida é chamada em uma sessão do SGBD, o carregador dinâmico carrega o arquivo objeto na memória para que a função possa ser chamada. O arquivo objeto carregado dinamicamente é mantido em memória após a primeira carga, ou seja, a primeira utilização de uma das funções contidas no código objeto. Chamadas posteriores das funções somente envolvem acessos à tabela de símbolos. Isto ocasiona um ganho no desempenho pelo fato das funções já estarem contidas na memória. A Figura 9 exibe os passos da execução de uma consulta aproximada utilizando as funções de similaridade `levenshtein` e `simNomes` implementadas.

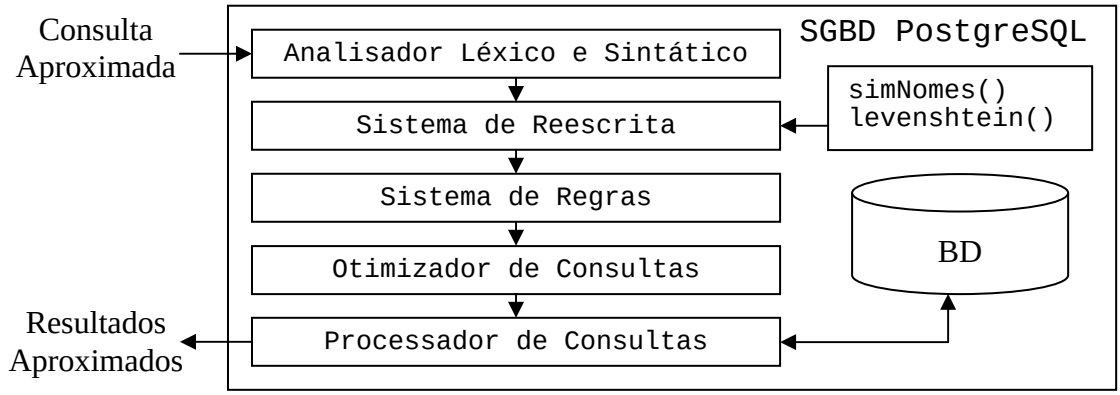

Figura 9. Execução de uma consulta utilizando as funções implementadas.

Após a execução do script SQL gerado, as tabelas `bdbcomp` e `dblp` contavam, respectivamente, com 3.976 e 529.202 instâncias. Uma consulta qualquer sobre o produto cartesiano de ambas as tabelas gera uma pesquisa sobre 3.976 x 529.202 linhas, ou seja, aproximadamente 2,1 bilhões de instâncias. Visando o cálculo da similaridade em tempo computacional não proibitivo, foram utilizadas as funções `simNomes` entre as iniciais dos autores e `levenshtein` sobre os títulos dos artigos científicos como filtros na seleção. A Figura 10 mostra a consulta realizada sobre a base de dados com o objetivo de identificar os casamentos que correspondem a versões de metadados provenientes das bibliotecas digitais em questão.

```
SELECT b.id, d.id, levenshtein(b.titulo, d.titulo)
FROM dblp d, bdbcomp b
WHERE simNomes(b.iniciais, d.iniciais, 0.75) = 1
AND levenshtein (b.titulo, d.titulo) > 0.75;
```

Figura 10. Consulta sobre a base de dados com o objetivo de identificar as versões dos metadados.

A consulta opera a função `simNomes` entre o produto cartesiano das duas tabelas (2,1 bilhões de instâncias) em aproximadamente 55 minutos e retorna em torno de 500

mil linhas (0,024% do tamanho total) para o cálculo da função `levenshtein`. O tempo total da transação é de 71 minutos. Foram detectados 874 pares de metadados que podem corresponder a versões do mesmo artigo científico, utilizando um limiar de 75% de similaridade tanto para a função `simNomes` quanto para `levenshtein`. Dentre os 874 pares detectados, apenas 4 não são versões do mesmo artigo científico, ou seja, não são pares relevantes para a consulta. Portanto, a precisão da consulta é de 99,54%. Já a revocação, não foi possível calculá-la, pois não é conhecido o número total de pares relevantes para a consulta. Entre as causas possíveis do baixo número de casamentos identificados podem estar:

- A maior parte das conferências indexadas pela BDBComp não são indexadas pela DBLP;
- A diferenciação entre maiúsculas e minúsculas causa um grande erro em ambas as funções de similaridade implementadas;
- Diferentes codificações para caracteres acentuados utilizados pelas bibliotecas digitais também provocam erros no cálculo da similaridade;
- Valores de limiar não ajustados corretamente.

Portanto, são necessários mais testes a fim de validar a abordagem utilizada, além de incluir o restante dos metadados na comparação.

5.2 Validando a proposta.

Experimentos comparativos são previstos a fim de validar a abordagem proposta. Deve-se comparar esta abordagem com outros trabalhos na área que não consideram versões e/ou proveniência dos dados.

6. Conclusões e trabalhos futuros

Atualmente, muitos usuários realizam buscas na Web sobre informações de artigos científicos, os quais estão dispostos em várias bibliotecas digitais. Um mesmo artigo científico pode ser referenciado por várias bibliotecas digitais, mas a representação desta referência é diferente para cada sistema. Devido à necessidade de identificar as diferentes representações de uma mesma publicação em sistemas de integração de bibliotecas digitais, foi proposto um mecanismo automático para detectar, representar, e consultar versões de objetos XML oriundos de bibliotecas digitais.

A abordagem proposta visa armazenar as versões dos objetos XML considerando informações de proveniência, de forma que seja possível realizar consultas sem perder as informações relativas à origem dos atributos dos objetos XML. Os resultados das consultas dos usuários são retornados com todos os dados possíveis que descrevem o artigo científico (cada parte da informação recuperada de uma determinada biblioteca digital) livres de redundância. Além disso, serão recuperadas informações de proveniência relativas ao local de origem destas informações.

Como trabalhos futuros destacam-se: (i) realizar mais testes com as funções de similaridade propostas. Calcular a precisão e a revocação das consultas a fim de medir o desempenho das funções de similaridade. Comparar os resultados obtidos com algumas abordagens clássicas na identificação de versões XML, a fim de validar o módulo de

detecção de versões. (ii) Especificar formalmente o modelo de proveniência de dados utilizado no módulo de armazenamento, aplicado ao formato XML. Logo, desenvolver o módulo de armazenamento. (iii) Desenvolver um protótipo funcional do sistema, composto pelos três módulos apresentados na arquitetura do sistema (Figura 4), com o objetivo de testar integralmente o mecanismo proposto. (iv) Comparar esta abordagem com outros trabalhos na área que não consideram versões e/ou proveniência de dados, validando assim a proposta.

Referências

Association for Computing Machinery (ACM). ACM Digital Library. Disponível em <http://portal.acm.org/dl.cfm>. Acesso: abril, 2007.

Baeza-Yates R.; Ribeiro-Neto, B. Modern Information Retrieval. ACM Press Series/Addison Wesley, New York, 1999.

Buneman, P.; Khanna, S.; Tan, W. Why and Where: A Characterization of Data Provenance. In International Conference on Database Theory, (2001).

Chawathe, S.; Garcia-Molina, H. Meaningful change detection in structured data. In International Conference on Management of Data, SIGMOD, pags. 26–37, Tucson, Arizona, Maio de 1997. ACM.

Dorneles, C.; Heuser, C; Lima, A; Silva A.; Moura, E.: Measuring similarity between collection of values. WIDM 2004: 56-63.

Flesca, S.; Manco, G.; Masciari, E.; Pontieri, L.; Pugliese, A. "Fast detection of XML structural similarity," Knowledge and Data Engineering, IEEE Transactions on , vol.17, no.2pp. 160- 175, Feb. 2005.

Flesca, S.; Manco, G.; Masciari, E.; Pontieri, L.; Pugliese, A. Detecting structural similarities between xml documents. In Proceedings of the Int. Workshop on The Web and Databases (WebDB 2002).

Giles, C.; Bollacker, K.; Lawrence, S. CiteSeer: An Automatic Citation Indexing System. In Proceedings of the 3rd ACM Conference on Digital Libraries (DL'98), pp 89-98, 1998.

Greenwood, M.; Goble, C.; Stevens, R.; Zhao, J.; Addis, M.; Marvin, D.; Moreau, L.; Oinn, T. "Provenance of e-Science Experiments - experience from Bioinformatics," in Proceedings of the UK OST e-Science 2nd AHM, 2003.

Institute of Electrical and Electronics Engineers, Inc. IEEE Computer Society Digital Library. Disponível em <http://www.computer.org/portal/site/csdl/>. Acesso: abril, 2007.

Joshi, S.; Agrawal, N.; Krishnapuram, R.; Negi, S. A bag of paths model for measuring structural similarity in Web documents, Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, August 24-27, 2003, Washington, D.C.

Laender, A.; Gonçalves, M.; Roberto, P. BDBComp: Building a Digital Library for the Brazilian Computer Science Community. In: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries, Tuscon, AZ, USA, pp. 23-24, 2004.

Lanter, D. "Design of a Lineage-Based Meta-Data Base for GIS," in Cartography and Geographic Information Systems, vol. 18, 1991.

Levenshtein, I. V. (1966). Binary codes capable of correcting deletions, insertions and reversals. Cybernetics and Control Theory, 10(8):707–710.

Lian, W.; Cheung, D.; Mamoulis, N.; Yiu, S. "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 1, pp. 82-96, Jan., 2004.

Nierman, A.; Jagadish, H. "Evaluating Structural Similarity in XML Documents". In Int'l Workshop on the Web and Databases (WebDB), Madison,WI, Jun. 2002.

Rahm, E.; Bernstein, P. A survey of approaches to automatic schema matching. Very Large Database J., 10(4):334–350, 2001.

Saccol, D; Nina, E; Galante, R. XML version detection. Relatório Técnico RP-355. Instituto de Informática, Universidade Federal do Rio Grande do Sul, 2007.

Simmhan, Y.; Plale, B.; Gannon, D. A survey of data provenance in e-science, ACM SIGMOD Record, v.34 n.3, September 2005.

Stasiu, R.; Heuser, C.; Silva, R. Estimating Recall and Precision for Vague Queries in Databases. CAiSE 2005: 187-200.

University of Trier. Digital Bibliography & Library Project (DBLP). Disponível em <http://dblp.uni-trier.de/>. Acesso: abril, 2007.

Van de Sompel, H.; Nelson, M.; Lagoze, C.; Warner, S. Resource Harvesting within the OAI-PMH Framework, D-Lib Magazine, December 2004, 10(12).

World Wide Web Consortium (W3C). Extensible Markup Language (XML). Disponível em <http://www.w3.org/XML/>. Acesso: abril, 2007.