Recuperação de Informação em Documentos XML: Uma Introdução

Emilio Mario Wieczorek

Mestrado em Ciência da Computação - Universidade Federal do Rio Grande do Sul

Porto Alegre, RS, Brasil

emilio@ufrgs.br

Abstract: *This paper aims to introduce the Information Retrieval (IR) in XML documents, showing some techniques of indexing and searching, with some languages used to query XML documents. Will be addressed as well, some Information Retrieval systems that work with XML documents. Furthermore, we introduce some efforts that have been developed in the area.*

Resumo: *Este artigo tem por objetivo introduzir a Recuperação de Informação (RI) em Documentos XML, demonstrando algumas técnicas de indexação e busca, apresentando algumas linguagens de consultas utilizadas para Documentos XML. Serão abordados também, alguns sistemas de Recuperação de Informação que trabalhem com Documentos XML. Além disso, serão apresentados alguns esforços que vem sendo desenvolvidos na área.*

1. Introdução

Com a crescente utilização da Internet para a propagação da informação, milhares de dados são criados diariamente (blog's, fórums, wiki's, etc), tornando a Internet o maior repositório de informações do globo. A partir disto, cada vez mais os usuários buscam estes dados na Internet e o armazenamento destas informações tem se tornado crucial para a indexação e busca correta destes dados. Surginge assim a necessidade de uma linguagem voltada tanto para armazenar a estrutura do documento quanto para o conteúdo das informações.

Este formato padrão para dados semi-estruturados na web o W3C (*World Wide Web Consortium*) criou um subconjunto apropriado da SGML, denominado XML (*eXtensible Markup Language*) [Baeza-Yates 1999]. A XML é uma linguagem que oferece uma abordagem padrão para descrição, captura, processamento e publicação de informações representadas por conteúdo, estrutura e apresentação. Recentemente, tornou-se um novo padrão para representação e troca de dados na Internet. Dessa forma, ela tem se tornado crucial para abordar a questão de como grandes coleções de documentos podem ser ordenados efetivamente, além de serem recuperados de forma eficiente.

A idéia base da recuperação de informação em documentos XML é conceber os documentos XML como uma coleção de documentos de texto com *tags* (marcações

encontradas em documentos XML, exemplo: <author></author>) adicionais e a relação entre estas *tags*.

Diante do apresentado, a motivação para a escolha do tópico esta no fato de a área de armazenamento e troca de informações pela Internet, principalmente em formato XML, estar crescendo consideravelmente nos últimos anos, tornando-se uma das áreas mais importantes de estudo quando se fala em busca na web, além de que, segundo [Carmel 2000], integrar Recuperação de Informação (RI) e técnicas de busca em arquivos XML permitirá buscas mais sofisticadas tanto na estrutura como no conteúdo destes documentos.

2. XML

XML é a linguagem de marcação de dados (*meta-markup language*) que provê um formato para descrever dados estruturados. Isso facilita declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas. A XML também vai permitir o surgimento de uma nova geração de aplicações de manipulação e visualização de dados via Internet, e difere da HTML em três grandes aspectos:

- Novas marcações podem ser definidas a vontade;
- Estruturas podem ser aninhadas a profundidades arbitrárias;
- Um documento XML pode conter uma descrição opcional de sua gramática.

A Linguagem XML permite que os usuários definam novas marcas para indicar estrutura. Diferente da HTML, um documento XML não fornece instruções sobre como deve ser exibido, sendo esta informação incluída separadamente em uma folha de estilo. Folhas de estilo são utilizadas para traduzir dados XML para HTML, podendo as páginas HTML resultantes ser exibidas por navegadores padrão.

Em sua forma básica, XML é uma sintaxe para transmissão de dados, e como tal, é bem provável que se torne um dos padrões principais para a troca de dados na Web. Para uma organização ou grupo de usuários, XML permite uma especificação que facilita a troca de dados e a reutilização por muitas aplicações.

A figura abaixo mostra um exemplo de arquivo XML.

```
<book>
      <publisher>
            <name>Morgan Kaufman</name>
      </publisher>
      <author> Author Name</author>
      <title>Title of Book</title>
</book>
```

Figura 1. Exemplo de arquivo XML

3. Busca em documentos XML

Os mecanismos de busca da Internet (por exemplo, Altavista, Google ou Infoseek) retornam milhares de documentos em uma simples consulta, alguns deles relevantes e outros não. A utilização da XML facilita a identificação da relevância de um documento por meio de suas *tags* em uma consulta, garantindo assim, maior precisão nos resultados da busca.

Robert Luk [Luk et al. 2000] analisou várias técnicas de indexação e busca para documentos XML. Através desta análise foram classificadas três abordagens: A abordagem orientada à banco de dados, a abordagem orientada à recuperação de informação e a abordagem híbrida.

3.1. Abordagem Orientada a Banco de Dados

Os dados indexados derivados de bancos de dados são convertidos para documentos XML que são traduzidos usando XSL2 (*eXtensible Stylesheet Language*). A abordagem orientada a banco de dados apresenta as seguintes vantagens: (i) as relações dos dados e integridade podem ser modeladas e checadas; (ii) as operações padrão ao banco de dados podem ser usadas; (iii) a linguagem de consulta é similar à linguagem padrão de banco de dados (SQL). A desvantagem está na dificuldade de se importar dados XML de um banco de dados e fazer mudanças em esquemas (schemas3) [Luk et al. 2000].

3.2. Abordagem Orientada à Recuperação de Informação

Na abordagem orientada à Recuperação de Informação, técnicas de RI são diretamente aplicadas para processar e recuperar documentos XML, onde cada documento XML é considerado um documento de texto com o excesso de *tags* adicionais. Vários métodos têm sido sugeridos para negociar com as *tags* XML. Estes incluem simplesmente descartar todas as *tags*; selecionar algumas *tags* importantes e inserí-las no índice; indexar todas as *tags* como se elas fossem termos de índice [Luk et al. 2000].

3.3. Abordagem Híbrida

Na abordagem híbrida, um método mais preciso é usado para especificar os elementos aninhados usando XPaths4 [XPaths]. A idéia básica é que listas invertidas de todos os termos e seus XPaths associados sejam indexados. Assim os resultados das buscas são mais satisfatórios, desde que a especificação dos XPaths limite os casamentos possíveis. A desvantagem é a necessidade do usuário possuir algum conhecimento adicional sobre XPaths para obter melhores resultados [Luk et al. 2000].

4. Linguagens de Consulta

Atualmente, a XML é o principal formato para armazenamento de dados semi-estruturados. Existem inúmeras propostas de Sistemas Gerenciadores de Bancos de Dados (SGBD) para dados semi-estruturados. Esses SGBD armazenam e manipulam dados em XML, criam índices, controlam concorrência no acesso a arquivos e processam consultas. Dados semi-estruturados possuem características que demandam linguagens de consulta específicas.

Existem várias linguagens de consulta para XML: XML-QL, XQL, XIRQL, Lorel, e a XSL [Kade 2000]. De acordo com [Kade 2000] as linguagens XML-QL e

XQL são semelhantes em termos de características, tais como: seleção e extração, ordenação, joins e consultas sobre a ordem.

4.1. XML-QL

XML-QL (*XML Query Language*) e a linguagem apoiada pelo W3C. Trata-se de uma extensão da linguagem SQL, com uma cláusula WHERE-CONSTRUCT, similar à cláusula SELECT-WHERE da SQL. Essa linguagem modela um documento XML através de um grafo [Coelho 2000].

Neste contexto, XML-QL tende a ser a linguagem padrão para XML, tratando os documentos XML como elementos de um banco de dados, realizando a extração, integração e a transformação dos dados para o formato XML. Além disso, é uma linguagem relacionalmente completa, sendo considerada simples de aprender e usar. XML-QL tem as seguintes características:

- É Declarativa, uma linguagem composta de relações entre os dados, de funções de correlação ou listas de declarações;
- É simples, tanto que técnicas de banco de dados conhecidas para otimização de consulta, estimação de custo, e re-escrita de consulta poderiam ser estendidas a XML-QL;
- Pode extrair dados de documentos XML existentes e construir novos documentos XML;
- Pode suportar ambas as visões ordenadas e não ordenadas em um documento XML (a implementação atual só suporta o modelo não ordenado).

A figura abaixo mostra um exemplo de uma consulta XML-QL utilizada para encontrar todos os autores que publicaram pela Morgan Kaufman.

```
WHERE <book>
          <publisher>
              <name> Morgan Kaufmann </>
          </>
          <author> $A </>
       </book> in "www.a.b.c/bib.xml"
CONSTRUCT <author> $A </>
```

Figura 2. Exemplo de consulta XML-QL

4.2. XIRQL

A linguagem XIRQL (*Extension of XQL for Information Retrieval*) [Fuhr 2000] combina os maiores conceitos de consultas XML com os conceitos de RI. Ela implementa características relacionadas à RI tais como [Fuhr 2000]:

- *Weighting*: Pesquisas em RI têm mostrado que a atribuição de pesos aos termos (*weighting*) no documento bem como a atribuição de pesos aos termos na

consulta são necessariamente ferramentas para recuperação efetiva em documentos textuais.

- *Data types and vague predicates*: Predicados vagos são a generalização natural de *weighting* para outros tipos de dados.
- *Relevance-oriented search*: Considere um documento estruturado, dividido em capítulos, seções e subseções. Para um documento uma subseção pode ser a melhor resposta para uma dada consulta do usuário, para outro pode ser o capítulo, ou seja, o cálculo de relevância está ligado à busca.
- *Semantic relativism*: XQL está quase presa à sintaxe XML, mas é possível usar XML com uma sintaxe diferente para expressar o mesmo tipo de significado.

XIRQL integra estas características por usar idéias de modelos de RI probabilísticos baseados em lógica, em combinação com conceitos da área de bancos de dados.

5. Sistemas de Recuperação de Informação para Documentos XML

Verificando a forte tendência da utilização de documentos XML para armazenamento e troca de informações na Internet, algumas empresas vêm desenvolvendo protótipos de sistemas comerciais para armazenamento e busca em documentos XML. Estes sistemas funcionam de forma a explorar semanticamente a estrutura de documentos XML, com o intuito de melhorar o retorno e a precisão de consultas nestes documentos. Três exemplos desses sistemas são o XYZFind [Egnor 2000] e o XMLFS [Azagury 2000].

5.1. XYZFind

Trata-se de um sistema para recuperação de informação estruturada usando XML. O XYZFind concentra-se em técnicas para explorar semanticamente a estrutura de documentos XML com o propósito de melhorar a precisão e o resultado das buscas. O XYZFind utiliza uma interface voltada para o usuário final (usuários que não são *experts* em esquemas de documentos XML).

O sistema utiliza um diálogo com o usuário a fim de elaborar uma consulta de forma estruturada. O diálogo é baseado num refinamento interativo em que o usuário, primeiramente, submete uma consulta apenas em texto, sem estrutura. Então, documentos que contêm todos os termos da consulta são procurados.

Um conjunto de esquemas representados pelo casamento (*matching*) dos documentos é gerado. O usuário então escolhe o esquema mais relevante à sua consulta e caso exista apenas um esquema possível não há a necessidade da escolha do usuário. Os esquemas podem ser rotulados amigavelmente se metadados estiverem disponíveis, tal como “Carros usados à venda”, caso contrário os esquemas são representados pelo nome do elemento raiz.

No próximo passo é dado ao usuário um formulário de pesquisa específico para o esquema selecionado, que possui entradas pertencentes ao esquema. Dependendo do tipo de dado (texto, número, etc) presente no corpo do XML os campos podem estar em

branco ou não. Com a consulta totalmente estruturada, ela é executada e o resultados são retornados ao usuário.

5.2. Xyleme

Xyleme é um *warehouse* dinâmico para dados XML da *Web* com suporte a consultas, controle de mudanças e integração dos dados. A característica principal do Xyleme é que este é baseado em *warehousing*.

A vantagem dessa característica é que uma consulta em páginas distribuídas na *Web* é realizada em cima dos dados armazenados. Outra vantagem é a possibilidade de notificar usuários quando páginas de seu interesse foram modificadas.

Os clientes Xyleme são executados em navegadores padrões usando *applets* Java [Aguilera 2000]. Uma consulta em Xyleme é mais que uma simples busca de palavras-chave, ela envolve uma análise da estrutura do documento e, em alguns casos, analisa informações vindas de vários documentos. A linguagem de consulta utilizada é uma extensão da OQL (*Object Query Language*) a qual fornece características de bancos de dados e Recuperação de Informação.

Para buscar páginas na *Web* e manter atualizado o repositório de dados foram implementados coletores (*crawler*) capazes de ler milhões de páginas por dia. Vários coletores são usados simultaneamente para fazer o trabalho. Apenas páginas em XML são armazenadas, enquanto as páginas em HTML são lidas para descobrir novos *links*.

Uma consulta em Xyleme é processada da seguinte forma:

1. Traduzida para uma expressão algébrica;
2. Contando com uma heurística a expressão é otimizada usando uma equivalência algébrica e alguma metainformação;
3. Um plano de execução distribuído, envolvendo índice de textos, é gerado e instalado;
4. O plano é avaliado;
5. A consulta é relaxada para retornar mais resultados.

Segundo [Xyleme 2001], as consultas em Xyleme são precisas, pois são formuladas através da estrutura de documentos e não através de palavras-chave. Xyleme fornece um mecanismo de visualização, capaz de habilitar o usuário a consultar uma estrutura única, que resume um domínio de interesse. Para classificar as DTDs (*Data Type Definitions*) em domínios (exemplo, turismo, comércio) são usadas análises estatíticas da similaridade entre palavras encontradas em diferentes DTDs e/ou documentos.

6. Esforços em Recuperação de Informação para Documentos XML

Desde 2000, quando aconteceu o primeiro workshop sobre XML e Recuperação de Informação, este tema tem estado presente nas conferências ACM SIGIR, mostrando o interesse da comunidade de Recuperação de Informação em explorar melhor as informações semi-estruturadas. Em 2002, na Finlândia, durante o 2° workshop sobre o

tema, debates sobre o sentido da recuperação de documentos XML, levaram a conclusão de que é importante tratá-la desvinculada de operações de Banco de Dados.

Através destas informações, serão apresentados aqui alguns esforços que vem sendo trabalhados com relação à Recuperação de Informação em Documentos XML, mostrando alguns caminhos e técnicas que vem sendo estudados na área.

6.1. Generating Vector Spaces On-the-fly for Flexible XML Retrieval

O foco dessa proposta é criar mecanismos que permitam ao usuário de uma máquina de busca compor consultas que explorem a estrutura dos documentos XML, obtendo resultados ordenados por relevância de acordo com uma granularidade desejada. Em busca deste objetivo, rompe as barreiras da máquina de busca tradicional que enxergam os documentos de forma plana e também as limitações das linguagens de consulta de XML que efetuam basicamente pesquisas exatas.

A ordenação por relevância leva em conta a estrutura do documento e as restrições que a consulta impõe sobre esta estrutura. Mais especificamente, os conteúdos em diferentes níveis da árvore que representam o documento terão importância diferente para consultas diferentes. Os autores utilizam a idéia de Fuhr [Fuhr 2000] que introduz pesos chamados de *augmentation weights* atribuídos a estatísticas como TFxIDF de cada termo, conforme a sua posição na árvore. Os elementos presentes na estrutura de cada documento são agrupados em nós de indexação que implementam as listas invertidas.

As consultas são agrupadas em três tipos diferentes, conforme o escopo da árvore que pretendem abranger:

- *Single_category* - consultas que buscam termos em apenas uma sub-árvore.
- *Multi-category* - consultas que buscam termos em mais de uma sub-árvore.
- *Nested- category* – consultas que buscam termos em toda uma sub-árvore mas cujos elementos possuem diferentes relevâncias para uma mesma consulta.

Para cada uma das três categorias de consulta os termos terão pesos diferentes, dependendo da estrutura da consulta. Estes pesos deverão ser calculados dinamicamente a partir de um índice básico pré-calculado.

Os índices básicos serão criados para cada elemento da árvore que possuem conteúdo textual, sendo que o artigo não detalha como isso ocorrerá, mas apenas sugere que poderão ser criados a partir de técnicas padrões de RI como extração de termos e eliminação de *stop words*. Partindo destes índices básicos, cada tipo de consulta terá uma fórmula especifica obtidas a partir do Modelo Vetorial.

$$RSV(e,q) = \sum_{t \in q} tf(t,e)\ ief_{cat}(t)^2\ tf(t,q)$$

Figura 1. Equação para Single-Category

$$RSV(e,q) = \sum_{t \in q} tf(t,e)\ ief_{mcat}(t)^2\ tf(t,q)$$

Figura 2. Equação para Multi-Category

$$RSV(e,q) = \sum_{se \in SE(e)} \sum_{t \in q} tf(t,se) \left(\prod_{l \in path(e,se)} aw_l \right)^2 ief_{cat(se)}(t)^2 \, tf(t,q)$$

$$= \sum_{se \in SE(e)} \left(\left(\prod_{l \in path(e,se)} aw_l \right)^2 \sum_{t \in q} tf(t,se) \; ief_{cat(se)}(t)^2 \, tf(t,q) \right)$$

Figura 3. Equação para Nested-Category

Em todas as equações observa-se que as estatísticas são calculadas para cada elemento na estrutura do documento e não para cada documento, como na fórmula original do Modelo Vetorial. Na equação da figura 2 a freqüência do elemento na coleção será a somatória da freqüência do elemento nas diversas categorias abrangidas pela consulta (*ief mcat*). Para a consulta aninhada (figura 3) a relevância será a soma ponderada da relevância para uma categoria simples, onde os pesos serão os *augmentation weights*

O grande avanço desta proposta é a flexibilidade admitida no processo de indexação, criando índices específicos para cada documento sem restrições em sua estrutura. Em contrapartida o processamento da consulta torna-se mais demorado tendo em vista a maior complexidade das fórmulas especialmente para consultas em multi-categorias e categorias aninhadas.

A intenção de tratar diferentes granularidades de consulta, por outro lado, exigem do usuário um conhecimento da estrutura do documento ou a sua dedução. Não há a possibilidade de uma busca integrada em documentos XML ou não, uma vez que os índices trazem estatísticas a nível de elemento e não de documento. Destina-se, portanto, exclusivamente à recuperação de informação em documentos XML.

6.2. An Extension of the Vector Space Model for Querying XML Documents via XML Fragments

Esta proposta busca criar uma ferramenta de pesquisa mais amigável, onde usuários possam expressar suas consultas na forma de texto livre ou através de consultas mais complexas dependendo do conhecimento que possuem das DTDs. O resultado também será ordenado por relevância colocando no topo do *ranking* documentos cuja estrutura mais se aproxime daquela proposta na consulta.

O *ranking* será gerado a partir de uma extensão do modelo vetorial que utilizará como unidades de indexação não apenas termos, mas pares da forma (ti,ci), onde os termos serão qualificados pelo contexto onde aparecem. Na equação da figura 4, o peso de termos individuais será substituído pelo peso dentro de uma contexto, wd(t,c). Mas, além disso, o modelo vetorial deixa de ter dimensões ortogonais entre t e c, passando a considerar diferentes níveis de semelhança entre os contexto da consulta e do documento, representada pelo fator cr(ci,ck), conforme equação da figura 5.

$$RVS(q,d) = \sum wq(ti)*wd(ti)/ |Q|*|D|$$

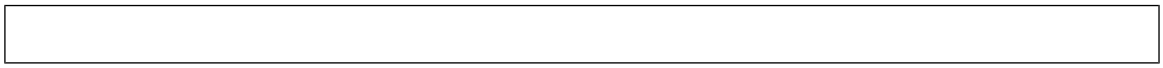

Figura4. Cálculodo pesopelo contexto

$$RVS(q,d) = \sum ci \sum ck\ wq(ti,ci)*wd(ti,ck)*Cr(ci,ck)/\ |Q|*|D|$$

Figura5. Cálculodiferentesníveis de semelhança

Durante a indexação os documentos XML serão percorridos e um vetor de pares (t,c) será extraído para criar o perfil de cada documento. Armazenando os termos e seus contextos, a lista invertida do termo t, contendo todos os documentos onde t aparece, será divida em diversas listas, uma para cada contexto, permitindo assim a recuperação do termo em determinado contexto.

A estrutura dos indexadores armazenará t e c como uma única chave t#c e no momento da recuperação o sistema poderá identificar ocorrências precisas de t dentro de um contexto c. Poderá, também, recuperar todos os contextos onde t aparece fazendo uma junção de todas as listas através do sufixo t#. As consultas poderão ser formuladas contendo os termos dentro de contextos ou apenas termos, como numa máquina de busca tradicional, ou poderá também conter uma mistura das duas situações, como por exemplo:

<article><title> XML tutorial <title><article> relating to XPath XQquery.

Gerando o seguinte conjunto (t,c) , para pesquisa:

{(xml,article/title), (tutorial, article/title), (relating, null), (Xpath, null), (XQuery, null)}

O Algoritmo de ordenação levará em conta o peso do termo no contexto, bem como a semelhança entre os contextos. O Cálculo de wd(t,c) é trivial e utiliza as estatísticas da chave t#c , t#c', etc. Para o cálculo de cr foram definidos alguns critérios e para cada um foi criado um fator.

Constata-se aqui um avanço ao permitir que sejam pesquisados documentos XML ou não. Há um tratamento dinâmico da semelhança entre os diversos contextos. Mas para que a estrutura dos documentos possam contribuir para a ordenação, as consultas deverão informar os elementos desejados, caso contrário todas as listas de um termo, em todos os contextos, serão aglutinadas constituindo-se assim um caso especial de consulta do Modelo Vetorial tradicional. Ou seja, a estrutura do documento não é avaliada para contribuir para a ordenação dos documentos a menos que o usuário tenha uma noção da estrutura dos mesmos e resolva explorar isso na formulação da consulta.

7. Conclusão

Já ha alguns anos a comunidade de bancos de dados semi-estruturados adotou XML como um formato padrão para troca de dados, fazendo com que vários esforços na área sejam desenvolvidos, tais como, modelos de consultas em documentos XML,

linguagens de consultas em documentos XML, algoritmos para busca e métodos de avaliação.

Quando falamos em recuperação de informação em documentos XML, temos que inferir que esta é uma área recente de estudo. Deste modo, há uma carência considerável de repositórios de informação em XML e, além disso, ainda não está claro quais aplicações adotarão XML como uma ferramenta para representação de informações. Antes de qualquer coisa, é preciso resolver problemas como o desenvolvimento de interfaces de consulta e obtenção de funcionalidade nos sistemas de busca em documentos XML.

Contudo, é fundamental avaliar o grande potencial que o armazenamento de informações em documentos XML possui, trabalhando na infra-estrutura das buscas neste formato.Um grande impasse envolve usuários e desenvolvedores de sistemas de busca em documentos XML, principalmente devido à falta de sistemas de busca eficientes, fazendo com que os usuários não invistam na criação de bancos de dados em XML. Conseqüentemente, os desenvolvedores de sistemas de busca também não investem na criação de tais sistemas.

Referências Bibliográficas

[Kade 2000] Kade, Adrovane Marques. **Uma interface visual para linguagem de consulta a dados semiestruturados**, 2000.

[Azagury 2000] Azagury, Alain; Factor, Michael e Mandler, Benny. **XMLFS: An XML-Aware File System**. IBM Research Lab in Haifa, 2000.

[Egnor 2000] Egnor, Daniel e Lord, Robert. **Structured Information Retrieval using XML**. http://www.xyzfind.com/ - Seattle, Washington, USA, 2000.

[Carmel 2000] Carmel, David; Maarek, Yoelle e Soffer, Aya. **XML and Information Retrieval**. SIGIR 2000. orkshop IBM Research Lab in Haifa, 2000.

[Robie 1998] Robie, Jonathan; Lapp, Joe e Schach, David. **XML Query Language (XQL)**. Disponível em: Disponível em http://www.w3.org/TandS/QL/QL98/pp/xql.html, 1998.

[Xyleme 2001] Lucie Xyleme. **A dynamic Warehouse for XML data of the Web**. IEEE Data Engineering Bulletin. 2001.

[Fuhr 2000] Fuhr, Norbert e GROßJOHANN, Kai. **XIRQL An Extension of XQL for Information Retrieval**. *Dortmund University, Germany*, 2000. Disponível em http://www.haifa.il.ibm.com/sigir00-xml/final-papers/KaiGross/sigir00.html

[Baeza-Yates 1999] Baeza0-Yates, Ricardo e Ribeiro-Neto, Berthier. **Modern Information Retrieval**. ACM Press / Addison-Wesley 1999.

[Luk et al. 2000] Luk, Robert; Chan, Alvin; Dollon, Tharam e Leong, H.V. **A Survey of Search Engines for XML Documents**. *Department of Computing, Hong Kong Polytechnic University*, 2000.

[McGrath 1999] McGrath, Sean. **XML Aplicações Práticas**. Rio de Janeiro. Campus, 1999.

[Coelho 2000] Coelho, Tatiana Almeida Souza. **Pocessamento Eficiente de Consultas XML Utilizando Arquivos Invertidos**, 2000. Disponível em: http://www.dcc.ufmg.br/pos/html/spg2000/anais/tatiana/tatiana.html

[Aguilera 2000] Aguilera, Vicent; Cluet, Sophie; Veltri, Pierangelo e Watez, Fanny. **Querying XML documents in xyleme**. ACM SIGIRWorkshop on XML and information retrieval, 2000.

[Grabs 2002] T. Grabs e H. J. Schek, "**Generating Vector Spaces On -the-fly for Flexible XML Retrieval**", ACM SIGIR 2002 Workshop on XML and Information Retrieval, Tampere, Finlândia, Agosto 2002.

[Carmel 2002] D. Carmel, N. Efraty, G. Landau, Y. Maarek e Y. Mass, "**An Extension of the Vector Space Model for Querying XML Documents via XML Fragments**", ACM SIGIR 2002 Whorkshop on XML and Information Retrieval, Tampere, Finlândia, Agosto de 2002.

[XPaths] http://www.w3.org/TR/xpath