Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Projeto Orientado a Computação I

Levantamento de Linguagens de Consulta para XML visando Análise Semântica

Orientadora: Mariza A. S. Bigonha
Aluna: Mírian Nunes Rubinstein

Relatório Final do Projeto Orientado I
Área: Linguagens de Programação

Av. Antônio Carlos, 6627
31270-010 - Belo Horizonte - MG
29 de Junho de 2000

Resumo

Incentivados pela idéia de garantir a qualidade da informação passada por documentos no formato eletrônico iniciamos este projeto de pesquisa. Trata-se de um estudo de linguagens de consulta para a linguagem XML, com o propósito de se escolher a linguagem mais adequada para a realização de restrições em estruturas sintáticas na linguagem SGML, as quais permitirão que esta linguagem possibilite a validação semântica automática. Nesta primeira fase realizamos o estudo; a respectiva caracterização das principais linguagens; e definimos a melhor linguagem, ou seja, aquela que possui o perfil que mais se adequa ao propósito. Após a seleção da linguagem de consulta mais adequada ao problema de garantia da correção semântica, a continuação deste trabalho consistirá na proposta de uma solução que permita que a análise semântica seja feita automaticamente em um documento.

1 Introdução

O crescimento desenfreado do número de publicações e do número de usuários fizeram da Internet uma grande fonte de informações heterogêneas disponíveis aos mais variados tipos de interesses e de necessidades. Em vista disto, a publicação de textos de qualidade, que não possuem erros ortográficos, erros sintáticos, ou erros semânticos é extremamente importante para proporcionar a confiança dos leitores e clientes.

Para atender a esta necessidade já existe no mercado vários editores que realizam a análise sintática garantindo a grafia correta das palavras do texto. Contudo, o suporte à análise semântica, por ser mais complexo, ainda não é feito com segurança de forma a garantir a correção das informações divulgadas. Por exemplo, a linguagem SGML, *Standard Generalized Markup Language*, [1] surgiu como um padrão internacional para publicação eletrônica. Muito embora ela permita validação estrutural automática, não há como controlar o conteúdo de um documento. Documentos estruturalmente corretos podem conter erros semânticos que acabam com a qualidade dos mesmos e comprometem a confiança dos leitores nas informações. Estes erros semânticos, na maior parte das vezes relacionados com a atribuição de datas erradas a eventos, atribuição de atos e sentenças a personagens históricos errados, etc, são comuns em vários textos, principalmente devido a falta de conhecimento ou atenção de revisores. Portanto, para evitar esses tipos de erros é imprescindível a aplicação de corretores semânticos.

Uma pergunta muito importante ao se falar em publicação eletrônica é: Como garantir a correção da informação contida em um documento? A busca da garantia da escrita de documentos com qualidade, estruturalmente e semanticamente corretos nos motivou e incentivou na proposta de realização deste projeto, focando na veracidade dos fatos e informações, uma vez que para a correção sintática já existem soluções.

Em relação a verificação semântica, encontramos na literatura a descrição de um sistema, o CCL[2], que além de realizar a publicação eletrônica de documentos, permite a associação de restrições semânticas aos elementos estruturais do documento via uma linguagem de restrição, fazendo com que muitos erros possam ser eliminados, conseqüentemente, aumentando-se a qualidade dos documentos. Na verdade este trabalho [2], no qual basearemos inicialmente nosso estudo, é uma extensão da linguagem SGML que propõe restrições sobre o conteúdo dos documentos com o objetivo de realizar a validação semântica automática. Note que é importante que as restrições semânticas sejam escritas em uma linguagem que seja uma extensão da linguagem padrão para se permitir a portabilidade de todo o patrimônio documental existente. No CCL, a linguagem de restrições escolhida é uma linguagem de consulta para XML. Como o XML é uma extensão da linguagem SGML, de modo geral, linguagens que realizam consultas possuem perfis adequados para este propósito.

XML, *Extensible Markup Language*[7], é definida como um "dialeto"extremamente simples da linguagem SGML, não é uma simples e pré−definida linguagem de marcação: é uma metalinguagem, ou seja, uma linguagem para descrever outras linguagens, a qual permite ao usuário criar seu próprio marcador.

Isto posto, o objetivo do nosso trabalho é apresentar uma opção, baseada nas abor-

dagens existentes [2],[3], que permita que seja realizada análise semântica automática de documentos, verificando a veracidade e correção dos dados apresentados.

Para atingir o objetivo proposto, este projeto foi decomposto em três etapas:

1. Levantamento dos pontos relevantes que uma linguagem de consulta deve possuir para possibilitar a análise semântica do conteúdo de um documento.

2. Estudo das linguagens de Consulta [3],[4],[5],[6] e outras, buscando as características delineadas no Item 1 e que possam ser úteis para garantir a boa qualidade e correção das informações contidas em um documento, escolhendo-se uma delas após uma análise crítica das abordagens pesquisadas.

3. Estudo detalhado da linguagem escolhida mostrando como esta linguagem deve ser aplicada para possibilitar a análise semântica. Especificamente será apresentado como se deve utilizar em um documento a linguagem XML e a linguagem de consulta escolhida, de forma a garantir a publicação de documentos com um mínimo de erros possíveis em relação ao seu conteúdo. E, se ainda houver erros, que a detecção das informações incorretas sejam feitas ainda em fase de edição do documento.

Este relatório apresenta os resultados obtidos após a conclusão dos Itens 1 e 2 e está organizado da seguinte forma. A Seção 2 apresenta o sistema CCL [2], que visa realizar a validação semântica de textos. Na Seção 3 é feito um levantamento das linguagens de consulta, utilizadas para elaborar as restrições para a linguagem XML. A Seção 4 mostra alguns fatores importantes na escolha de uma linguagem para realização de consultas visando validação semântica, e faz um estudo comparativo entre as abordagens pesquisadas e aquela apresentada em [2]. Baseado no resultado desta comparação concluimos mostrando na Seção 4.4 a linguagem escolhida para se fazer a análise semântica.

Após a seleção da linguagem de consulta mais adequada ao problema em mãos, a continuação deste trabalho consistirá em mostrar via exemplos, como se deve proceder para que a análise semântica seja feita automaticamente em um documento, como exposto na Seção 5.

2 O Sistema CCL

A edição SGML trouxe por si só um aumento na qualidade da produção documental, fazendo com que a validação estrutural/sintática dos documentos no momento da sua edição fosse possível. No entanto, como exposto anteriormente, isso não é suficiente para garantir a qualidade de um documento, dado que a correção semântica também é um fator muito importante.

Procurando preencher esta lacuna, o sistema CCL[2] tem como foco a correção da informação. Ele propõe a utilização do SGML como meio de controle da qualidade na produção documental, propondo uma extensão que permite exprimir restrições sobre o conteúdo dos documentos de modo a possibilitar alguma validação semântica.

A metodologia proposta pelo CCL permite controlar a estrutura semântica do documento, introduzindo restrições no sistema por meio de um conjunto de valores pré-definidos, colocadas como comentários especiais que obedeçam certas condições, as quais permitirão a verificação da semântica do documento. Essas restrições são associadas ao DTD, *Document Type Definition*. O DDT é a parte mais importante de um documento SGML, é onde está definida a estrutura do documento. O critério é o seguinte: uma linha de comentário é colocada no ínicio do DTD, contendo uma referência a um arquivo, onde as restrições aparecem. O funcionamento é muito simples, em um processamento de texto normal, o SGML ignora as restrições, uma vez que as mesmas não passam de comentários comuns, mantendo, então, a compatibilidade. Já o processador especial, responsável pela validação semântica, lê esta sessão de comentários e a processa.

Optou-se, neste sistema, pela utilização de linguagens do tipo XSLT[15], pois estas vão além da simples seleção, elas permitem um segundo nível de seleção, baseado em restrições sobre conteúdo. Este segundo nível de seleção é utilizado para especificar as restrições, de tal forma que alterando-se a sua semântica ao invés de retornar um conjunto de elementos, é retornado um valor booleano. Quando um valor falso da condição é encontrado, uma mensagem de erro é emitida.

A linguagem de restrição escolhida pertence ao conjunto de linguagens de consulta para XML, que é uma extensão da linguagem SGML, mantendo a compatibilidade. Ela é dividida em duas partes: (1)uma linguagem para a seleção de elementos e (2)uma linguagem para especificar as restrições.

Mais detalhes sobre este sistema podem ser encontrados em [2].

3 Linguagens de Consulta

3.1 Introdução

Uma linguagen de consulta é uma linguagem para extração e reestruturação de conteúdo de documentos. Quando se trata de um documento XML surge uma pergunta: Porque não utilizar as linguagens de consulta padrão para documentos relacionais ou orientados por objeto? Essas linguagens não se aplicam diretamente a XML, pois os dados de um documento XML diferem dos dados de documentos desse tipo. Um dado XML está mais próximo de um modelo de dados que tem sido estudado atualmente: o modelo de dados semi-estruturado. Neste modelo, os dados são irregulares: conceitos semelhantes são representados utilizando tipos diferentes, faltam alguns dados, conjuntos heterogêneos estão presentes e a estrutura dos objetos não é totalmente conhecida.

Existe na literatura várias linguagens de consulta para documentos que usam o modelo de dados semi-estruturado, entre elas [11],[10],[3],[5],[12]. Estas linguagens atendem às principais características que uma linguagem de consulta deve possuir para que consultas possam ser realizadas: auto poder de expressão, bom desempenho e compatibilidade com outras linguagens. Contudo, o desenvolvimento de uma linguagem de consulta padrão para XML ainda está no seu início. Mostraremos nas próximas seções algumas das linguagens

de consulta para XML, dando ênfase às características mais relevantes de tais linguagens.

3.2 XSL

A linguagem XSL, *Extensible Stylesheet Language*[11], possui características que podem servir como base para uma linguagem de consulta para XML[3]. O padrão XSL é simples; e possui uma sintaxe concisa, que possibilita a realização de consultas poderosas. Seu propósito é identificar um subconjunto de um documento XML baseado em cadeias de ancestrais, *wildcards* [1] e qualificadores, tais como valores de atributos, a fim de que as consultas possam ser realizadas.

Um programa XSL consiste em um conjunto de regras de *templates* [2], onde cada regra consiste em duas partes: um padrão o qual é comparado com nodos no documento a ser consultado e um *template*, que é instanciado para formar parte da árvore resultado. XSL trabalha bem sobre dados regulares e repetitivos e também provê formas de tratar dados irregulares e recursivos tais como os que existem em documentos do cotidiano.

As principais características de XSL que a torna uma linguagem apropriada para várias aplicações XML são:

Modelo de dados: todas as linguagens de consulta possuem um modelo de dados para se abstrair da representação física dos dados. Por exemplo, linguagens de consulta relacionais operam sobre relações e linguagem de consulta orientadas a objetos operam sobre objetos. XSL opera sobre documentos utilizando o modelo definido na especificação do Xpath[8]. Um documento XML é modelado como uma árvore que contém sete tipos de nodos: nodos raiz, nodos de elementos, nodos de texto, nodos de atributos, nodos de *namespace*[3], nodos de processamento de instruções e nodos de comentários[3]. Para cada tipo de nodo, existe uma forma de determinar um valor, string, que pode ser parte do nodo ou pode ser computado dos valores dos descendentes.

Expressões de caminho: quando estamos realizando uma consulta sobre dados semi-estruturados, principalmente quando não se conhece a estrutura, uma boa prática é utilizar uma espécie de consulta navegável baseada em expressões de caminho. Em XSL expressões de caminho definem localizações relativas e absolutas. Um caminho de localização relativa consiste em um ou mais passos, nodos XML, separados pelo operador filho, "/", ou pelo operador descendente, "//". Uma localização absoluta tem um "/"ou um "//"seguida, opcionalmente, por um caminho de localização relativa.

[1] Um símbolo é utilizado para representar vários tipos. Por exemplo, quando estamos procurando por um arquivo e não sabemos a seu extensão, podemos procurar por nome.* onde "*"é um wildcard, e representa todos os arquivos com nome "nome"e qualquer extensão.

[2] Esqueleto, modelo, a ser seguido para se construir algo.

[3] Namespaces em XML suportam a criação de dados cuja estrutura é definida por múltiplos esquemas. Por exemplo, assuma que endereços podem ser definidos de acordo com múltiplos DTD's, ou seja, um para os EUA e outro para a Europa, entre outros.Nós podemos qualificar um elemento pelo *namespace* no qual ele foi definido.

Construindo novos elementos: um novo elemento XML pode ser criado por meio de mecanismos de construção de uma linguagem de consulta. Em XSL, os novos elementos do resultado de uma consulta são especificados por meio dos nomes de seus *tags* dentro das regras do template.

Operações de conjunto: XSL suporta operações de união e negação, mas não existe interseção explícita.

Consultas Aninhadas: em XSL as templates podem ser aninhadas.

3.2.1 Avaliação da Linguagem XSL

As principais vantagens em se utilizar XSL são a capacidade de transformação, a sua definição formal, a modularidade da sintaxe padrão, o seu mecanismo único para modelar um documento XML e apresentar resultados em *templates*. Além disso, ela acomoda tanto dados regulares como irregulares e recursivos e mais importante, é um trabalho em progresso pela W3C[9].

3.3 XML-QL

Como alternativa para uma linguagem de consulta para XML foi proposto pela W3, *World Wide Web Consortium*, a XML-QL[10]. Esta linguagem tem características da linguagem SQL[3], como pode ser visto na construção SELECT-WHERE, e características de linguagens de consulta recentemente desenvolvidas para dados semi-estruturados. XML-QL aplica a experiência acumulada no desenvolvimento e implementação dessas linguagens para realizar consultas em documentos XML utilizando casamento de padrões[3]. XML-QL pode expressar consultas, para extrair dados de documentos XML, assim como realizar transformações, as quais podem, por exemplo, mapear dados de diferentes DTDs[2], que é essencialmente uma gramática, cuja funcionalidade é restringir as *tags* e a estrutura do documento XML, e podem integrar dados XML de diferentes campos. As principais características de XML-QL são:

Modelo de dados: para XML-QL permite uma variação de modelo de dados semi-estruturados. Em XML-QL, um documento XML é modelado por um grafo XML. Neste grafo, nodos representam identificadores de objetos, denominados OID, vértices são rotulados com identificadores de elementos, nodos intermediários são rotulados por conjuntos de valores-atributos representado atributos e as folhas são rotuladas com valores. Cada grafo tem um nodo denominado raiz.

Expressões de caminho: em XML-QL expressões de caminho são permitidas em qualquer lugar que exista um elemento XML. Elas permitem operações de alternação e concatenação, semelhantes àquelas usadas em expressões regulares. As variáveis de *tag*, as quais permitem que uma variável seja ligada a qualquer elemento em um grafo XML, e as expressões de caminho regulares tornam possível escrever uma consulta

que pode ser aplicada a dois ou mais campos de dados XML possuindo DTDs semelhantes, mas não idênticos. Isso ocorre porque estas expressões podem tratar a variedade de formato de dados de múltiplos campos.

Construindo novos elementos: em XML-QL a reestruturação é especificada na cláusula *construct*, a qual contém os novos *tags*, constantes e variáveis do novo documento.

Operações de conjunto: XML-QL suporta as operações de união e interseção, mas não suporta operação de diferença.

Consultas Aninhadas: como XML-QL é baseada em SQL, as consultas podem estar aninhadas a qualquer nível.

3.3.1 Linguagem XML-QL X Linguagem XSL

XSL é orientada primeiramente para especificar estilo e laioute de documentos XML. XML-QL compartilha algumas funcionalidades com XSL[10], mas XML-QL suporta operações mais intensivas, como junções e agregações, e tem um melhor suporte para construir novos dados XML, essencial para realizar transformações. XSL modela um documento XML como uma árvore. XML-QL modela um documento XMl como um grafo.

3.4 XML-GL

XML-GL[3] é uma linguagem lógica, baseada em grafos e de propósito geral para realização de consultas sobre documentos com dados semi-estruturados. Como XML-GL é uma linguagem de consulta gráfica, ela conta com uma representação para documentos XML e seus DTDs por meios de grafos XML rotulados. Todos os elementos de XML-GL estão dispostos visualmente. Ela é considerada uma linguagem de interface agradável. As suas principais características são:

Modelo de dados: é utilizado um modelo de dados gráfico(XML-GDM) para representar ambos, DTDs e documentos propriamente ditos. XML-GDM é utilizado também para formular consultas. Elementos XML são representados por retângulos e as propriedades como círculos. Arestas entre nodos representam relação de conteúdo ou referência.

Expressões de caminho: é permitido se percorrer o grafo XML-GL, procurando um elemento a qualquer nível de profundidade.

Construindo novos elementos: um novo elemento é construído por meio de uma nova caixa de elementos, nomeada arbitrariamente. Estes podem ser adicionados e nomeados arbitrariamente.

Operações de conjunto: em XML-GL as consultas permitem múltiplos grafos no lado direito da consulta, isto permite expressar a operação de união. Negação permite expressar a operação de diferença e interseção pode ser construída por repetidas operações de negação.

Consultas Aninhadas: Não são suportadas por XML-GL.

Uma consulta XML-GL é feita em quatro partes:

- Extração: parte que identifica o alvo da consulta, indicando o(s) documento(s) alvo(s) e os elementos alvo dentro destes documentos.
- Casamento(opcional): parte que identifica características especificas que os elementos selecionados devem possuir. Semelhante a cláusula *WHERE* do SQL.
- Construção: especifica a estrutura do documento resultado. Uma mesma consulta pode ser escrita com a parte de construção diferente de maneira que os resultados sejam formatados diferentemente.
- Clip(opcional): especifica os elementos do documento alvo que estarão presentes no documento resultado.

3.4.1 Avaliação da Linguagem XML-GL

A principal característica que difere XML-GL das demais linguagens de consulta para XML é que as consultas são formuladas visualmente utilizando um formalismo baseado em grafos bem próximo a estrutura de um documento XML, quando se compara a representação visual de documentos XML oferecidas por ferramentas de autoria. Entretanto XML-GL, não é uma interface visual convencional, mas sim uma linguagem de consulta baseada em grafos, com ambos, sintaxe e semântica definidos em termos de estruturas de grafo e operações.

3.5 XQL

XQL[5] é uma notação para selecionar e filtrar os elementos e texto de um documento XML[11]. XQL foi desenvolvida especificamente para documentos XML. Trata-se de uma linguagem de consulta de propósito geral, com a característica de ser, sintaticamente, muito simples e compacta. A linguagem XQL foi muito inspirada pela linguagem XML-QL, Seção 3.3. Essa linguagem tem características comuns às linguagens de consulta SQL e OQL. XQL provê, também, uma extensão natural para a linguagem XSL. Ela possui a capacidade de identificar classes de nodos, via lógica booleana; filtros; indexação para coleção de nodos; e etc. XQL foi desenvolvida para ser usada em muitos contextos. É uma notação para retirar informações sobre documentos. A informação pode ser um conjunto de nodos; informação sobre relações; ou valores derivados. A especificação não indica o formato de saída. O resultado de uma consulta pode ser um nodo, um documento XML, um arranjo, ou alguma outra estrutura. Suas principais características são:

Modelo de dados: enquanto os desenvolvedores de várias das linguagens de consulta para XML apresentadas neste texto introduziram seu próprio modelo de dados explicitamente, XQL trabalha sobre as características do modelo de dados da linguagem XML[10]. Os projetistas da linguagem assumiram o modelo de dados da linguagem XML, realçando que cada nodo tem um tipo e um conteúdo ou um valor; que relações entre nodos podem ser hierárquicas (pai/filho, ancestral/descendente), posicional(absoluta, relativa); e seqüencial (precede, precede imediatamente)[3].

Expressões de caminho: expressões de caminho definem localizações relativas e absolutas. Um caminho de localização relativa consiste em um ou mais passos, nodos XML, separados pelo operador filho, "/", ou pelo operador descendente, "//". Uma localização absoluta tem um "/"ou um "//"seguida, opcionalmente, por um caminho de localização relativa.

Construindo novos elementos: em XQL nenhum novo elemento pode ser adicionado aos que já existem, visto que não existe nenhum mecanismo de construção.

Operações de conjunto: XQL suporta união e interseção. A operação de negação permite que XQL tenha o poder de expressar a operação de diferença.

Consultas Aninhadas: XQL não suporta consultas aninhadas.

Coerção de tipos: o conceito de coerção está relacionado à conversão automática de tipos para satisfazer um contexto. Trata-se de uma característica muito importante, visto que auxilia o usuário a medida que permite que este realize consultas sem se preocupar com o tipo preciso dos objetos o que é apropriado quando se trata de dados semi-estruturados. Em XQL, dois valores podem ser comparados somente depois de realizar o *casting* [4] explícito dos mesmos, assim como ocorre em linguagens de programação tradicionais.

3.5.1 Avaliação da Linguagem XQL

Jonathan Robie of Texcel, um dos co-autores da linguagem, ressaltou alguns de seus aspectos significantes :"ela permite que se realize consultas sobre documentos XML como em um banco de dados; ela tem um modelo que é diretamente baseado nos relacionamentos encontrados em um documento XML, os quais permitem consultas poderosas serem expressas de maneira simples; ela é utilizada facilmente para realizar um implementação simples, mas também é possível que se use para aplicações de larga escala".

O fato desta linguagem possuir a característica de coerção de tipos é importante visto que esta operação permite que consultas sejam realizadas sem se preocupar com o tipo preciso dos objetos, o que é apropriado quando se trata de dados semi-estruturados semelhante ao modelo de dados da linguagem XML.

[4] Conversão entre tipos

3.6 Lorel

A linguagem Lorel[12] foi desenvolvida para realizar consultas sobre dados semi-estruturados. É uma linguagem de fácil compreensão, tem uma sintaxe familiar *select-from-where* e é baseada em OQL[13], com certas modificações e extensões que são úteis quando se quer realizar consultas sobre dados semi-estruturados[12]. A linguagem Lorel, possui coerção de tipos, o que é muito importante quando se trata de dados semi-estruturados, e possui eficientes expressões de caminho, importante quando detalhes da estrutura não são conhecidos pelo usuário. Suas principais características são:

Modelo de dados: no modelo de dados em Lorel, um elemento XML é um par <eid,value>, onde eid é um identificador de elementos, e value é um valor atômico ou um valor complexo; seguido por uma lista ordenada de pares de atributos, nome e valor atômico, que representam os atributos XML; seguidos por uma lista ordenada de pares de atributos <eid,value> chamada *crosslink elements*, representando os atributos dos IDREF XML[8]; seguidos por uma lista ordenada de pares de atributos <eid,value>, chamada de sub-elementos normais, que representam os relacionamentos de conteúdo de XML. Existe uma representação por grafos do modelo de dados, onde os nodos correspondem aos elementos de dados do XML e os vértices são designados tanto como *crosslinks elements* , como vértices normais. Vértices são rotulados. Cada grafo XML possui um ou mais nodos, designados como pontos de entrada.

Expressões de caminho: em Lorel, expressões de caminho são poderosas e flexíveis, elas admitem vários *Unix-Like wildcards*. Cada expressão de caminho deve ter um contexto, o elemento raiz do documento.

Construindo novos elementos: em Lorel, o novo elemento é construído chamando a função xml () com três parâmetros, o tipo, o rótulo e o valor (ou valores), dados explicitamente através do OID, ou dados implicitamente especificando-se para a consulta para gerá-lo .

Operações de conjunto: Lorel suporta operações de união, diferença e interseção.

Consultas aninhadas: em Lorel, que é inspirada em SQL, as consultas podem estar aninhadas a qualquer nível.

Coerção de tipos: em Lorel, comparações entre objetos ou valores são orientadas para "o mais intuitivo", ao invés de retornar um erro de tipos, quando comparamos objetos ou valores de tipos diferentes. Existem regras de coerção para os mais variados tipos e os predicados ou funções correspondentes. Então temos que a comparação entre objetos atômicos, objetos complexos ou conjuntos de objetos é aceitável quando existe uma interpretação óbvia.

3.6.1 Avaliação da Linguagem Lorel

Apesar das semelhanças entre o modelo de dados semi-estruturado e XML, existem algumas diferenças que forçaram algumas modificações na linguagem para que ela suporte XML. O modelo de dados original da linguagem, OEM, *Object Exchange Model*,[12] e características da linguagem de consulta, como expressões de caminho, entre outras características, sofreram alterações para atingir o objetivo de tornar a linguagem uma linguagem de consulta eficiente para XML.

3.7 Avaliação das Linguagens de Consulta

Depois da breve descrição das principais linguagens de consulta para XML, podemos fazer uma avaliação, citando algumas características particulares de cada linguagem. A linguagem XSL já é um trabalho em progresso pela W3C[9]; tem a capacidade de transformação; possui gramática formal; modularidade da sintaxe padrão; acomoda tanto dados regulares como irregulares e recursivos; e tem um mecanismo único para modelar um documento XML e apresentar resultados em templates. A linguagem XML-QL compartilha algumas funcionalidades com XSL, mas XML-QL suporta operações mais intensivas, como junções e agregações, e tem maior suporte para construir novos dados XML, que é essencial para realizar as transformações. XML-GL difere de outras linguagens de consulta a medida que nesta linguagem as consultas são formuladas visualmente utilizando um formalismo baseado em grafos bem próximo a estrutura de um documento XML, quando se compara a representação visual de documentos XML oferecidas por ferramentas de autoria. Entretanto XML-GL, não é uma interface visual convencional, mas uma linguagem de consulta baseada em grafos com ambos, sintaxe e semântica definidos em termos de estruturas de grafo e operações. XQL permite que consultas sejam realizadas sobre documentos XML como em um banco de dados; ela tem um modelo que permite que consultas poderosas possam ser expressas de maneira simples; ela tem grande facilidade de uso para realizar implementações simples; e também pode ser usada para aplicações de larga escala. Apesar das semelhanças entre o modelo de dados semi-estruturado e XML, existem algumas diferenças que forçaram algumas modificações na linguagem Lorel para que ela suporte XML.O modelo de dados original da linguagem, OEM, *Object Exchange Model*,[12] e características da linguagem de consulta, como expressões de caminho, entre outras características, sofreram alterações para atingir o objetivo de tornar a linguagem uma linguagem de consulta eficiente para XML.

Na nossa opinião, a linguagem XQL é a que apresenta características mais apropriadas ao objetivo deste projeto devido a sua sintaxe simples e compacta e a sua facilidade de uso. Maiores detalhes sobre a linguagem escolhida serão apresentados na Seção 4.4.

4 Fatores Importantes na Escolha da Linguagem de Consulta

4.1 Introdução

O SGML, linguagem de anotação (veja Seção 1) permite validação semântica de um documento, ou seja, permite controlar a sua estrutura, mas não controla o conteúdo do documento. Para que certas características semânticas dos documentos sejam preservadas, é preciso que se adicionem restrições ao DTD. Partindo do pressuposto que a linguagem SGML é um padrão internacional, se quisermos usá-la, devemos modificá-la, e para realizar modificações sobre ela, um fator muito importante deve ser levado em consideração: a compatibilidade. Isto posto, uma possível solução seria adicionar restrições ao DTD, de acordo com o CCL, conforme exposto na Seção 2.

4.2 A Linguagem de Restrições

O que se observou sobre as restrições que devem ser adicionadas para permitir validação semântica[2] é que estas devem ser bem simples. Geralmente deve-se restringir o valor de alguns elementos atômicos a um determinado domínio; verificar relações entre elementos; e verificar a existência de um elemento. Observou-se também, que as restrições devem ser associadas a um elemento em um determinado contexto. Isso se deve ao fato de que um elemento pode aparecer em contextos diferentes, e portanto, possuir semânticas diferentes. Necessita-se, então, de uma linguagem que realize uma seleção de contextos e especifique as restrições.

Concluiu-se que a realização da seleção de contextos poderia ser executada por uma linguagem de consulta semelhante às linguagens necessárias em banco de dados tradicionais como aquelas descritas na Seção 3.1. Mais ainda, como a compatibilidade deve ser um quesito importante, esta linguagem poderia ser escolhida dentro de um conjunto de linguagens de consulta para XML, que é uma extensão da linguagem SGML.

Optou-se, então, por realizar o estudo sobre linguagens do tipo XSLT[15], pois estas vão além da seleção, permitindo um segundo nível de seleção, baseado em restrições sobre conteúdo, conforme mostrado na Seção 2.

A seleção de conteúdo e a condição, restrição propriamente dita, seriam escritas dentro de uma mesma sintaxe. Dessa forma, a linguagem pode selecionar elementos dentro de um contexto e calcular condições sobre estes elementos. A seguir mostramos um exemplo de uma consulta e como podemos encará-la como uma restrição:

- Selecionar todos os elementos jornalista que tenham um e-mail.
  Se fôssemos encarar esta consulta como uma consulta comum, teríamos como resultado um conjunto de jornalistas que tivessem e-mail. Do ponto de vista das restrições, o resultado seria uma mensagem de erro para todos os jornalistas que não tivessem um e-mail.

4.3 Como Restringir a Escolha de uma Linguagem de Consulta

Do estudo realizado, foram observadas várias características desejáveis em uma linguagem para que ela possa implementar as restrições. São elas:

Deve ser declarativa no lugar de procedimental: uma linguagem de consulta é declarativa quando, na especificação da consulta, é definido o conteúdo do resultado ao invés de uma forma para computá-lo.

Deve ter operadores para a especificação de filtros: o resultado de uma consulta pode ser refinado via uma sub-consulta, restrição aplicada ao resultado da consulta principal. A sub-consulta é equivalente a cláusula *Where do SQL*. O resultado da aplicação de uma sub-consulta é um valor booleano e os elementos que obtiverem um valor verdadeiro estarão presentes no resultado final.

Deve prover uma maneira fácil para a realização da seleção de contexto: um contexto de procura é o conjunto de nodos nos quais se irá aplicar a expressão para calcular o resultado. Todos os nodos do contexto de procura são filhos do mesmo nodo pai, que juntamente com seus atributos e os atributos do nodo pai, constituem o contexto.

Deve permitir operações lógicas sobre conteúdo: quando um elemento deve atender a uma ou/e outra característica.

Deve adicionar restrições sobre o conteúdo do documento.

O próximo passo é definir qual linguagem de consulta que melhor se adequa às características importantes para se realizar as restrições.

4.4 A Linguagem Escolhida

Considerando as características desejáveis em uma linguagem de consulta para a validação semântica expostas na Seção 4.3 e as linguagens apresentadas na Seção 3, concluímos que a linguagem de escolha é XQL por possuir um perfil mais adequado para a realização das restrições.

A seguir apresentamos as características consideradas mais importantes, as linguagens que as possuem e o que nos levou à escolha de XQL.

- Linguagem Declarativa

  De acordo com a definição de linguagem declarativa, Seção 4.3, todas as linguagens pesquisadas são declarativas. Lorel parece-se com OQL e portanto uma linguagem *calculus-based*. XML-QL tem uma variável unificada como um dos seus ingredientes e então lembra uma linguagem lógica, mas com um peculiar estilo XML. XML-GL lembra QBE[3]. Ambos XSL e XQL fazem uso do padrão *URL-like*[3].

- Operadores para especificação de filtros

  A linguagem XQL realiza a consulta de maneira concisa, breve. Esta tem um padrão de navegação com a condição de filtro. O filtro é existencialmente quantificado. O resultado é, por convenção, armazenado em um elemento chamado de *xql: result*.

- Maneira fácil de selecionar o contexto

  A seleção do contexto pode ser realizada através de expressões de caminho. Todas as linguagens pesquisadas suportam, pelo menos parcialmente, a especificação de expressões de caminho.

- Permite operações lógicas sobre o conteúdo

  A única linguagem que permite parcialmente a realização de operações lógicas sobre o conteúdo é a linguagem XML-QL. Todas as demais linguagens permitem a realização total de tais operações.

- Adiciona restrições sobre o conteúdo do documento

  Todas as linguagens apresentadas permitem ter um segundo nível de seleção baseado em restrições sobre o conteúdo do documento.

Embora a maioria das linguagens possua características semelhantes, temos como diferencial os seguintes aspectos:

A semântica do resultado de uma consulta na linguagem Lorel é um conjunto de OIDs [3], enquanto as outras linguagens produzem um documento XML, o que é mais interessante, quando estamos preocupados em escrever as restrições.

A facilidade de uso também é um fator importante no momento de se escrever as restrições. Esta propriedade indica o quão fácil é, para o programador escrever ou ler uma consulta. Nesta perspectiva, a linguagem XML-GL é fácil de ler e entender quando associada a uma interface gráfica. A linguagem XQL é também simples e fácil, menos poderosa no entanto, mas estas duas linguagens são as que possuem maior facilidade de uso.

Considerando a compatibilidade temos que a linguagem XQL foi desenvolvida especificamente para documentos XML. Ela possui o modelo de dados semelhante ao modelo de dados da linguagem XML, o que implica na facilidade de escrita das restrições. O fatores compatibilidade, simplicidade e facilidade de uso foram os que realmente pesaram na escolha de XQL.

5 Conclusão

Concluímos, diante do exposto neste documento, que a linguagem XQL é a linguagem mais adequada para escrever as restrições. Apesar de seu poder de expressão ser limitado, em razão da falta da operação de *join* e da construção de resultado, isto não é considerado um fator relevante no momento da construção das restrições. XQL é uma linguagem de

consulta de propósito geral, com a característica de ser, sintaticamente, muito simples e compacta. Ela adiciona operadores para a especificação de filtros, operações lógicas sobre o conteúdo, indexação em conjunto de elementos, e restrições sobre o conteúdo dos elementos. XQL é, portanto, uma notação para especificação de operações de extração de informação de documentos estruturados.

A continuação deste trabalho, previsto para o final do próximo semestre, consistirá na apresentação de uma solução para que a análise semântica seja feita automaticamente em um documento.

Referências

[1] Goldfarb, C. P. *The SGML Handbook. Oxford*

[2] Ramalho, J. C., Henriques, P. R. *CCL: Content Constrain Language - Linguagem para Especificaçào de Restrições em Documentos SGML/XML. Departamento de Informática - Escola de Engenharia - Universidade do Minho - Portugal, 2000*

[3] Bonifati, A., Ceri, S. *Comparative Analysis of five XML Query Languages. Dipartimento di Elettronica e Informazione, Politecnico di Milano*

[4] DeRose, S. *XQuery: A unified syntax for linking and querying general XML documents. The Query Language Workshop, December 5,1998*

[5] Robie, J., Lapp, J., Schach, D. *XML Query Language (XQL). The Query Language Workshop, December 5,1998*

[6] Tompa, F. *Providing Flexibe Access in a Query Language for XML. The Query Language Workshop, December 5,1998*

[7] Frequently Asked Questions about the Extensible Markup Language. *(http://www.ucc.ie/xml/), June, 1999*

[8] *http://www.w3.org/TR/xpath*

[9] http://www.w3.org

[10] XML-QL: A Query language for XML. *(http://www.w3.org/TR/1998/NOTE-xml-ql-19980819), August, 1998*

[11] XML Query Language(XQL).*(http://www.w3.org/Style/XSL/Group/1998/09/XQL-proposal.html)*

[12] *(http://www-db.stanford.edu/lore)*

[13] R.G.G.Cattell *The Object Database Standard: ODMG-93. Morgan Kaufmann, San Francisco, Califórnia, 1994*

[14] XML.Com. *(http://www.xml.com)*

[15] XSL transformation(XSLT)- version 1.0 - *(http://www.w3.org/TR/1999/REC - XSLT - 19991116.html) World Wide Web Consortium Recommendation 16 - November- 1999*

Mírian Nunes Rubinstein
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Tel.: (031)499-5860 / (031)499-5842
Fax.: (031) 499-5858

Prof. Mariza Andrade da Silva Bigonha
Professor Adjunto
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Tel.: (031)499-5860 / (031)499-5891
Fax.: (031) 499-5858