XChange: Compreensão de Mudanças em Documentos XML

Guilherme Martins[1], Celio Larcher Junior[1], Alessandreia Oliveira[1,2], Leonardo Murta[2], Vanessa Braganholo[2]
[1]Universidade Federal de Juiz de Fora
[2]Universidade Federal Fluminense
{guilherme, celio}@ice.ufjf.br,
{alessandreia, leomurta, vanessa}@ic.uff.br

Abstract. *Web Applications are increasingly using XML documents nowadays. Those documents evolve over time, so managing these changes becomes fundamental. The focus of existing research for comparing XML documents resides in identifying syntactic changes, which is not always enough. This paper presents the XChange approach to support the evolution of XML documents based on inference, using Prolog. Differently from existing approaches, XChange uses the syntactic changes in two versions of an XML document, which usually have a common purpose, to infer the reason of the changes.*

Resumo. *Documentos XML têm sido utilizados cada vez mais no contexto de aplicações Web. Um problema relacionado é que os documentos XML evoluem ao longo do tempo e gerenciar estas mudanças torna-se fundamental. Abordagens existentes têm seu foco na identificação de mudanças sintáticas para a comparação de versões de documentos XML, o que nem sempre é suficiente. Diante disto, este artigo apresenta a abordagem XChange para apoiar a evolução de documentos XML baseada em inferência, usando a linguagem Prolog. Diferente das abordagens existentes, XChange utiliza as mudanças sintáticas de duas versões de um documento XML, que usualmente têm um propósito em comum, para inferir a razão das mudanças.*

1. Introdução

Cada vez mais, é possível perceber a crescente utilização de documentos XML tanto na área de desenvolvimento quanto na troca e no armazenamento de informações [Moro *et al.* 2009]. Com isso, surge a necessidade de gerenciá-los, o que não é uma tarefa trivial. Tal dificuldade se deve, por exemplo, ao fato destes documentos apresentarem uma estrutura hierárquica e marcadores definidos pelo usuário. Apesar de permitir flexibilidade na representação dos dados, isto dificulta o acompanhamento de sua evolução, principalmente em grandes repositórios de dados.

Na literatura este problema já vem sendo estudado há algum tempo [Cobena *et al.* 2002; Wang *et al.* 2003]. Porém, o foco dessas abordagens está na detecção de mudanças sintáticas entre as versões de um documento XML e, em alguns casos, isso não é suficiente. De fato, existem situações onde não basta detectar o que mudou, mas também é necessário inferir a razão das modificações. Por exemplo, num cenário onde um documento XML representa o cadastro de funcionários de uma empresa, pode ser necessário verificar o cargo de um funcionário num dado momento no passado ou

quando determinados funcionários foram contratados ou demitidos. Se essa informação não existir explicitamente no documento, a sua percepção é dificultada se abordagens de detecção de mudanças puramente sintáticas forem utilizadas.

Diante disso, este trabalho apresenta o XChange [Oliveira *et al.* 2012], que usa inferência para apoiar a compreensão das mudanças entre versões de um documento XML. As versões são transformadas em fatos Prolog e, a partir de um conjunto de regras de inferência, pode-se deduzir a intenção do usuário ao criar a segunda versão. Enquanto uma abordagem puramente sintática apresentaria um conjunto de elementos adicionados ou removidos, o XChange visa extrair desse conjunto o significado de alto nível da mudança, facilitando a sua compreensão pelos especialistas do domínio.

Este trabalho está organizado como a seguir. A Seção 2 descreve alguns trabalhos relacionados a esta proposta. A Seção 3 apresenta a abordagem XChange, enquanto a Seção 4 descreve a sua implementação. Para finalizar, a Seção 5 apresenta as considerações finais e algumas sugestões de trabalhos futuros.

2. Comparação de Documentos XML

Existem iniciativas relacionadas à proposta deste artigo, que foram identificadas e classificadas em três contextos: detecção de mudanças em páginas Web frequentemente acessadas, mineração de mudanças em documentos XML e *diff* de documentos XML.

Existem ferramentas com o intuito de detectar mudanças em páginas Web frequentemente acessadas. WebVigiL [Chamakura *et al.* 2005] é um sistema de monitoramento de mudanças para páginas Web escritas em XML e HTML. CX-Diff [Jacob *et al.* 2005], por outro lado, é uma abordagem específica para detectar mudanças de conteúdo de *tags* de documentos XML. Por fim, o algoritmo heurístico proposto por Lim e Ng [2004], tem como objetivo descobrir mudanças entre dois arquivos XML ou HTML, hierarquicamente estruturados e representados por uma árvore ordenada. Estas abordagens analisam os aspectos sintáticos, não considerando a semântica das modificações dos documentos XML, como proposto pelo XChange.

Em relação à mineração de dados em documentos XML, a proposta de Zhao et al. [2006] é baseada na identificação de estruturas que se alteram frequentemente através de uma abordagem denominada mineração de delta estrutural. O objetivo é extrair informações através de sequências de mudanças estruturais no documento XML. Rusu et al. [2006] e Sonawane e Tambe [2013] utilizam regras de associação extraídas de versões de um documento XML para realizar previsões sobre alterações futuras do documento. Estas propostas estão mais concentradas na descoberta de regras de associação ou de estruturas que mudam com frequência e não na razão das mudanças.

Algumas propostas procuram descobrir a sequência de operações necessárias para transformar uma versão anterior de um documento XML em sua versão atual. Wang et al. [2003] propõem o algoritmo X-Diff que procura atingir a minimalidade da sequência de operações e usa árvores não ordenadas. O algoritmo Xy-Diff [Cobena *et al.* 2002] foi proposto para realizar a detecção de diferenças entre versões distintas de documentos XML. Sundaram e Madria [2012] apresentam uma abordagem de detecção de mudanças entre versões de documentos XML não ordenados, armazenados em um banco de dados relacional. Estas abordagens têm seu foco no *diff* sintático ao invés de se preocuparem com a semântica das mudanças, como XChange propõe.

3. Visão Geral do XChange

Esta seção apresenta o XChange (http://gems.ic.uff.br/xchange) para apoiar a evolução de documentos XML. A partir do processamento de duas versões de um documento XML, é possível compreender as mudanças ocorridas. Para isso, são utilizadas as informações explícitas associadas a cada versão para deduzir conhecimento implícito sobre as mudanças. Esta dedução é realizada através de um conjunto de regras.

A Figura 1 apresenta o XChange. Os dados de entrada são constituídos por duas versões de um documento XML, de domínio qualquer, que são pré-processados e transformados em um conjunto de fatos Prolog, e por um conjunto de regras definidas por um especialista de domínio no início do processo. Estes dados de entrada são utilizados pelo motor de inferência Prolog, que, ao aplicar as regras sobre os fatos, retorna o significado de alto nível da modificação (razão da evolução do documento XML, de v1 para v2), que é a principal contribuição deste trabalho. Maiores detalhes sobre a abordagem estão disponíveis em Oliveira *et al*. [2012].

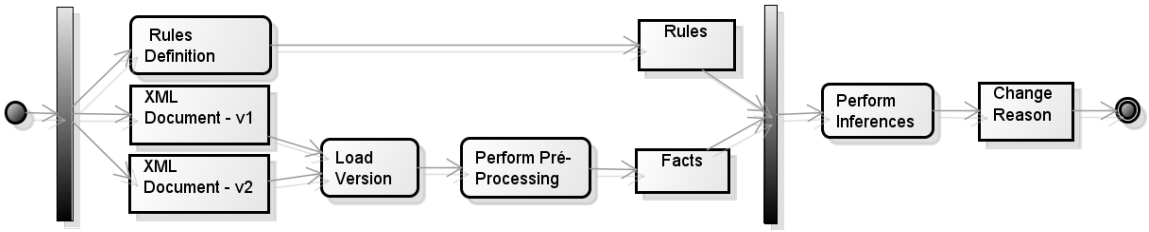


Figura 1: XChange [Oliveira *et al.* 2012]

4. Detalhamento Técnico do XChange

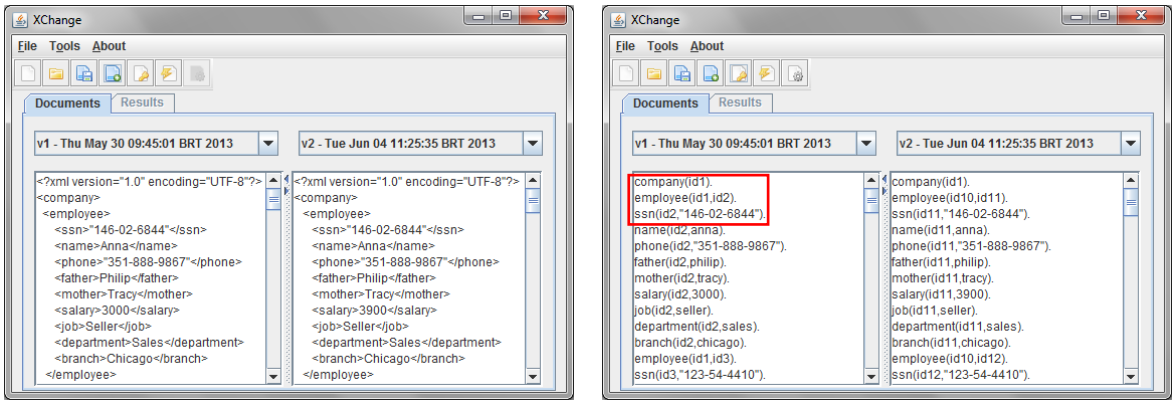
XChange foi desenvolvido, utilizando a linguagem Java, para criar um ambiente conciso onde o usuário pode monitorar as mudanças em diferentes versões de um documento XML. A biblioteca tuProlog [Denti *et al.* 2001] foi incorporada para apoiar a inferência. A abordagem pode ser dividida em três etapas que são descritas a seguir: geração de fatos Prolog, definição das regras e inferência de dados.

4.1. Geração de fatos Prolog

Para se realizar inferências lógicas sobre as informações contidas nas versões do documento XML, é necessário que estas sejam transformadas em fatos Prolog. Para isto foi utilizada a abordagem proposta por Lima *et al*. [2012]. O processo de tradução gera vários fatos Prolog a partir de um único documento XML, transformando elementos em predicados e seus conteúdos em constantes.

No cenário de cadastro de funcionários (Figura 2), o processo de tradução é realizado em 3 passos [Lima *et al.* 2012]. O primeiro traduz a raiz (*<company>)* em um fato com o nome do elemento e argumento único igual a um identificador gerado para estabelecer o vínculo com seus elementos filhos (*company* (id1)). Para relacionar predicados distintos, são criadas constantes Prolog, que atuam como identificadores que preservam a ligação pai/filho entre os elementos XML correspondentes. O segundo passo traduz os elementos simples sem atributos (por ex., *<ssn>146-02-6844</ssn>)*. O resultado é um fato com nome igual ao nome do elemento e argumentos iguais ao identificador do elemento pai e o conteúdo do elemento corrente (*ssn (id2, "146-02-6844")*. O terceiro passo é a tradução dos elementos complexos. Um novo identificador

é criado para relacionar os filhos ao pai. O resultado é a geração de um fato com o nome do elemento e dois argumentos: o identificador do elemento pai e um novo identificador gerado para referenciá-lo (por ex., *employee* (id1, id2)). XChange estende o método de tradução original [Lima *et al.* 2012], para utilizar duas versões de um documento XML (v1 e v2) como entrada (Figura 2.a) e gerar os fatos correspondentes (Figura 2.b).

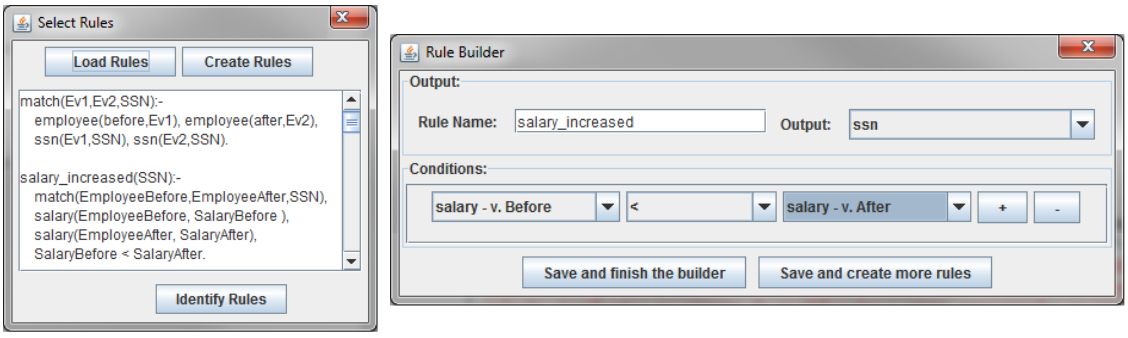


(a) Documento XML (v1 e v2) (b) Fatos Prolog (v1 e v2)

Figura 2: XChange

4.2. Definição de Regras

Um especialista com conhecimento no domínio pode, no início do processo, criar regras que permitam a inferência sobre fatos Prolog. XChange oferece a possibilidade de se criar regras ou carregar regras previamente. A Figura 3.a apresenta duas regras, previamente definidas, no cenário do cadastro de funcionários. A regra *salary_increased* analisa duas versões do documento XML (v1 e v2) e identifica os funcionários que receberam um aumento de salário. A regra auxiliar (*match*) é usada para vincular elementos de v1 e v2. A regra *match*, neste cenário, utiliza o SSN (equivalente ao CPF nos EUA) como um identificador (chave) para estabelecer a correspondência entre elementos de diferentes versões de um documento XML. Esta estratégia garante a unicidade do elemento *employee* e deve ser usada também em outros cenários/domínios (nesse caso, com chave diferente, dependendo do domínio).



(a) Regras (b) Criação de Regras

Figura 3. Criação de Regras

O XChange utiliza uma interface gráfica para facilitar a geração das regras pelo usuário. Com o módulo de criação de regras ativado, é solicitada a seleção do identificador para estabelecer a correspondência entre os elementos nas diferentes versões. Escolhida a chave (neste cenário, *ssn*), o próximo passo é a definição da regra (Figura 3.b). É necessário informar o nome da regra (*salary_increased*), a saída (*ssn*) e

as condições a serem satisfeitas *(salary – v.Before < salary – v.After)*. Um conjunto inicial, envolvendo os elementos e relação entre eles, é fornecido para a definição das condições. Os demais elementos que compõem a regra e se referem à identificação de elemento correspondente são inseridos automaticamente após a definição da regra.

4.3. Inferência de dados

Após a transformação das versões de um documento XML em fatos Prolog e da definição das regras, a máquina de inferência opera sobre os fatos combinando-os com as regras. Esta operação aplica cada regra definida aos fatos gerados, obtendo seus respectivos resultados. Com isso, é possível identificar o motivo da evolução de um documento XML a partir de suas versões (de v1 para v2). Para exemplificar, a Figura 4 mostra o SSN dos 4 funcionários que receberam aumento. O primeiro *ssn* mostrado corresponde à funcionária *Anna*. Como pode ser visto na Figura 2.a, *Anna* realmente recebeu aumento (*<salary>3000</salary>* em *v1 e <salary>3900</salary>* em *v2)*.

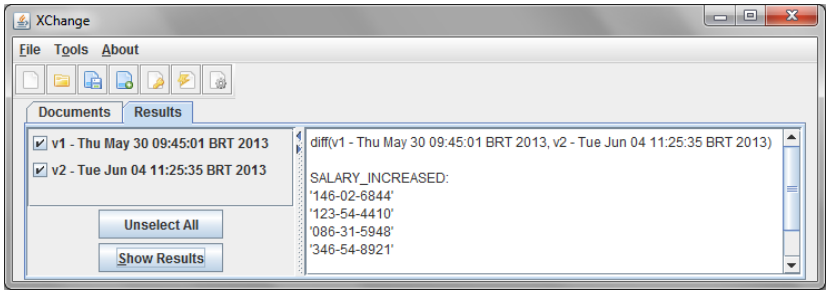


Figura 4. Resultados

5. Considerações Finais

Este trabalho apresentou o XChange, desenvolvido para viabilizar a identificação da razão real das modificações em documentos XML, tomando como base a análise das modificações sintáticas granulares em atributos e elementos. Para isto, o XChange usa um mecanismo de inferência baseado na linguagem Prolog e, a partir de um conjunto de regras e de fatos Prolog, pode-se inferir a razão das mudanças. A principal contribuição deste trabalho é, portanto, apoiar a compreensão de mudanças sobre documentos XML. Em documentos XML grandes, que passaram por muitas modificações entre suas versões, pode haver uma redução significativa no número de modificações a serem apresentadas ao usuário ao migrar de modificações sintáticas para semânticas. Essa relação pode permitir que centenas de modificações sintáticas sejam transformadas em dezenas de modificações semânticas. Essa característica tem potencial para contribuir com a facilidade de análise e compreensão sobre a evolução de documentos XML.

Além disso, esta abordagem não exige que o usuário seja especialista em Prolog, uma vez que utiliza uma interface gráfica que gera consultas Prolog a partir de uma seleção de opções em um alto nível de abstração, facilitando o processo de consulta. Por fim, as regras geradas são válidas para todos os documentos pertencentes ao mesmo esquema, o que faz com que essa etapa ocorra uma única vez para um dado esquema.

Uma limitação da abordagem atual é a identificação automática de elementos correspondentes. Hoje, no contexto do exemplo de utilização, o usuário utiliza a regra *match* (Figura 3.a) para identificar elementos correspondentes em v1 e v2. Esta regra usa um atributo-chave (SSN no exemplo). Dependendo da forma como os documentos

XML são gerenciados, não há garantia de que o valor permanece o mesmo entre as versões (por ex., pode acontecer um erro de digitação no valor do SSN na v2). Para atenuar esse problema, uma abordagem baseada em análise de similaridade [Dorneles *et al.* 2009] está sendo desenvolvida. Outros aspectos a serem explorados: uso desta proposta na combinação de versões e na geração automática de regras semânticas.

Agradecimentos

Os autores agradecem ao CNPq e à FAPERJ pelo apoio financeiro.

Referências

Chamakura S., Sachde A., Chakravarthy S., Arora A. WebVigiL: Monitoring Multiple Web Pages and Presentation of XML Pages. In Data Engineering, 2005.

Cobena G., Abiteboul S., Marian A. Detecting changes in XML documents. In International Conference on Data Engineering (ICDE), IEEE, p. 41–52, 2002.

Denti E., Omicini A., Ricci A. tuProlog: A light-weight Prolog for Internet applications and infrastructures. Pr. Asp. Declar. Lang.: 184–198, 2001.

Dorneles C. F., Nunes M. F., Heuser C. A., Moreira V. P., Silva A. S. da, Moura E. S. de. A strategy for allowing meaningful and comparable scores in approximate matching. Inf Syst 34(8): 740–756, 2009.

Jacob J., Sachde A., Chakravarthy S. CX-DIFF: a change detection algorithm for XML content and change visualization for WebVigiL. Data Knowl. Eng. 52(2), 2005.

Lim S., Ng Y.-K. Change Discovery of Hierarchically Structured, Order-Sensitive Data in HTML/XML Documents. In Symposium on Applications and the Internet (SAINT), 2004.

Lima D., Delgado C., Murta L., Braganholo V. Towards querying implicit knowledge in XML documents. J. Inf. Data Manag. JIDM: 51–60, 2012.

Moro M. M., Braganholo V., Dorneles C. F., Duarte D., Galante R., Mello R. S. XML: Some papers in a haystack. SIGMOD Rec.: 29–34, 2009.

Oliveira A., Murta, L., Braganholo V. Uso de Inferência na Compreensão das Modificações em Documentos Semiestruturados. In Simpósio Brasileiro de Banco de Dados (SBBD), 2012.

Rusu L. I., Rahayu W., Taniar D. Mining changes from versions of dynamic XML documents. Knowl. Discov. XML Doc. KDXD: 3–12, 2006.

Sonawane V., Tambe S. Extracting interesting konowledge from versons of dynamic XML documents. IJRET Int. J. Res. Eng. Technol. 2(4), 2013.

Sundaram S., Madria S. K. A change detection system for unordered XML data using a relational model. Data Knowl. Eng. 72: 257–284, 2012.

Wang Y., DeWitt D. J., Cai J.-Y. X-Diff: an effective change detection algorithm for XML documents. In Intern. Conf. on Data Engineering (ICDE), p. 519 – 530, 2003.

Zhao Q., Chen L., Bhowmick S. S., Madria S. XML Structural Delta Mining: Issues and Challenges., 2006.