Uma Interface Gráfica para Algoritmos de Detecção de Diferenças entre Documentos XML

Frantchesco Cecchin, Carmem Satie Hara

Departamento de Informática – Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brasil

{frantchesco,carmem}@inf.ufpr.br

Trabalho de Graduação

Abstract. *There are many algorithms in the literature for detecting differences between XML documents. However, only a few provide a graphical interface to display their results. Such a tool allows the user to clearly identify the differences and check the correctness of the result. As a consequence, the verification step becomes faster and less prone to errors. The purpose of this work is to design and implement a graphical interface that interprets and presents in a clear way the differences found by the execution of a diff algorithm. It displays two versions of an XML document, emphasizing the modifications detected by a diff algorithm. The tool supports three diferent algorithms, allowing the user to choose which algorithm to execute, and presents the results in a uniform way.*

Resumo. *Existem diversos algoritmos voltados à detecção de diferenças entre documentos XML, comumente chamados de algoritmos de diff para XML. No entanto, poucos possuem uma interface gráfica que facilite a visualização de tais diferenças. Assim, quando se tem à disposição uma ferramenta que permite ao usuário visualizar mais claramente as diferenças entre dois documentos XML, o trabalho de verificação torna-se mais rápido e menos suscetível a erros. Pensando nesse propósito, foi elaborada uma interface gráfica, chamada VisualX, que interpreta e apresenta de forma clara as diferenças encontradas pela execução de um algoritmo de diff para XML. Sua interface apresenta duas versões de um documento XML, enfatizando, através de cores, as modificações detectadas pelo algoritmo. A ferramenta dá suporte a três algoritmos distintos. Ou seja, o usuário pode escolher qual dos algoritmos deseja utilizar e o resultado é apresentado de maneira uniforme pela ferramenta.*

1. Introdução

O uso da *Linguagem de Marcação Extensível* (XML - *eXtensible Markup Language*) tem sido cada vez mais difundida para troca de dados na *Web*. Com a XML tornando-se um padrão para publicação e transporte de dados via *Web* faz-se necessário o desenvolvimento de algumas ferramentas que facilitem o trabalho dos usuários dessa linguagem. Dentre estas ferramentas destacam-se os algoritmos de detecção de diferenças entre documentos XML, aqui chamados apenas de algoritmos de *diff*.

A função essencial de um algoritmo de *diff* é encontrar um *edit script*, ou delta, entre duas versões de um documento de tempos `t(i-1)` e `t(i)`. Este *edit script* representa as mudanças entre as duas versões, ou seja, dada a versão `t(i-1)` e o *edit script*, é

possível chegar à versão `t(i)` [Cobéna et al. 2002]. Como os dados publicados na *web* sofrem constantes alterações, ferramentas de *diff* ganham uma importância singular, pois através destas os usuários e administradores de base de dados semi-estruturados podem monitorar e controlar alterações entre versões destes documentos. Entre outros benefícios que o gerenciamento de modificações em documentos XML pode trazer podem ser citados:

- *Auxílio no exame de processos editoriais*, permitindo aos editores verificar as mudanças feitas nos dados originais;
- *Automatização de sincronização de mudanças* feitas em sistemas diferentes, combinando as atualizações com o arquivo XML original;
- *Redução do custo de transmissão*, através do envio apenas dos dados modificados;
- *Redução dos custos de armazenamento*, através do armazenamento exclusivo do delta para posterior reconstrução da fonte original.

A obtenção de um delta "semanticamente" correto, ou seja, com operações de atualização que reflitam a expectativa do usuário é muito importante se o mesmo for utilizado para a atualização incremental de um repositório de dados no qual a versão anterior do documento está armazenado. Como exemplo considere a seguinte situação. Dois funcionários, Maria e João, trocam de salas de forma que Maria que utilizava a sala `r20` passa a ocupar a sala `r30` e João se muda da sala `r30` para `r20`. Um algoritmo de *diff* pode detectar que as mudanças no documento ocorreram ou no valor das salas dos funcionários ou nos ocupantes das salas. Embora ambas as alternativas reflitam as alterações ocorridas no documento, apenas uma delas corresponde à expectativa do usuário e pode ser considerada "semanticamente correta". Assim, se o documento contém dados de funcionários, a alternativa correta é na modificação das salas. Por outro lado, se o documento trata da alocação física, a alteração nos ocupantes seria a mais desejável. Somente operações de atualização corretas podem ser utilizadas para incrementalmente atualizar de forma coerente um repositório integrado de dados. Um dos objetivos da ferramenta descrita neste artigo é facilitar a interpretação e verificação dos resultados gerados por diferentes algoritmos de *diff*.

O restante do artigo está dividido da seguinte forma: na Seção 2 é apresentada uma visão geral sobre algoritmos de *diff* para documentos XML. A Seção3 apresenta algumas ferramentas existentes que auxiliam a verificação de diferenças entre documentos XML. Na Seção 4 é apresentada a interface desenvolvida, sua implementação, objetivos e relevância desta para a análise de diferenças entre documentos XML. Por fim, na Seção 5 são feitas as considerações finais e indicações de trabalhos futuros.

2. Algoritmos de *diff* para XML

Os usuários não se interessam somente pelos valores atuais dos dados, mas também pelas mudanças [dos Santos 2006]. Para atender essa necessidade foram desenvolvidos vários algoritmos de detecção de diferenças entre documentos XML. Por possuírem objetivos similares, os algoritmos apresentados possuem fortes conexões uns com os outros. Porém, cada algoritmo tem seu foco em determinado aspecto da detecção de diferenças e procura atender otimizações ou adaptações sobre algum outro previamente desenvolvido.

Os algoritmos de *diff* consistem de dois passos. No primeiro, elementos em ambas as versões são "casados" de acordo com algum critério. O critério mais comumente

utilizado é o de similaridade e, no caso de XML, o caminho da raiz até o elemento. No segundo passo, o *edit script* é gerado baseado no conjunto de casamentos realizados: elementos casados podem ter sofrido atualização de valor; elementos da primeira versão que não foram casados são considerados removidos e elementos da nova versão que não foram casados são considerados inseridos. Os algoritmos podem ser avaliados pela sua complexidade e também pelo tamanho do *edit script* gerado, ou seja, a quantidade de operações que o compõe. Porém, este último pode ser considerado um critério "sintático".

Embora uma quantidade pequena de operações do *edit script* garanta que a versão antiga seja transformada na nova versão de forma eficiente, ela não garante que o *edit script* contenha o resultado semântico esperado. Este resultado depende, em grande parte, dos casamentos realizados. Ou seja, quanto mais o algoritmo casar elementos em ambas as versões que correspondem à mesma entidade do mundo real, melhor será a qualidade do *edit script* gerado. Esta qualidade é essencial na propagação de atualizações baseadas no resultado do algoritmo de *diff*. Ou seja, se o documento original encontra-se armazenado e o sistema deseja atualizá-lo de acordo com as diferenças detectadas, estas operações devem ser realizadas sobre os elementos corretos.

Geralmente, os algoritmos de detecção de diferenças não oferecem uma interface amigável para a visualização do *edit script* gerado, dificultando a análise do mesmo pelos usuários. Além disso, os algoritmos existentes diferem em seus propósitos. Existem algoritmos projetados para comparar grandes volumes de dados de forma eficiente, como o XyDiff [Cobéna et al. 2002], bem como aqueles projetados para gerar resultados semanticamente significativos, como o XKeyMatch [dos Santos 2006]. A ferramenta proposta neste trabalho procura atender estas necessidades oferecendo uma interface adequada para visualização das modificações e a possibilidade de comparar resultados obtidos por diferentes algoritmos de *diff*.

3. Trabalhos Relacionados

Existem diversas ferramentas de visualização de diferenças entre documentos XML. O XML Tools [Studio 2007] é uma ferramenta proprietária que faz parte de uma *suite* comercial. Este pacote consiste de um conjunto de ferramentas para trabalhar com XML, serviços Web, publicações XML e outras tecnologias relacionadas com XML. A ferramenta mostra o documento no formato de árvore de expansão, utilizando ícones para representar que houve alguma modificação, remoção ou adição em um elemento não expandido. Isto auxilia a visualização de documentos XML extensos. Por outro lado, ele trabalha apenas com um algoritmo de detecção, que é proprietário, e está disponível apenas para o sistema operacional Microsoft Windows. Outra ferramenta comercial interessante é a ExamXML [A7Soft 2007], que permite a criação de regras para a comparação, como por exemplo ignorar elementos que possuem um atributo com determinado nome. Usando estas regras o usuário pode comparar somente os elementos que considerar relevante. Esta ferramenta também utiliza um algoritmo de *diff* proprietário e não oferece portabilidade.

O DiffDog [Altova 2007] é outra ferramenta que possui uma interface simples que serve para comparar arquivos, diretórios e, especialmente, documentos XML. A ferramenta apresenta os documentos em formato de texto, realçando as diferenças, e cria uma ligação visual entre os elementos casados em uma versão e outra do documento XML.

Essa funcionalidade é bastante interessante, mas a forma que a interface faz as ligações pode se tornar incompreensível quando existem muitas diferenças entre os documentos.

Por último, o XSDelta [Perini et al. 2006] disponibiliza uma interface que expõe aos administradores de bases de dados semi-estruturadas todas as operações envolvidas na evolução de um esquema XML. A funcionalidade principal dessa ferramente é a conversão de esquemas DTD para esquemas XML Schema. Também permite a visualização, em formato texto, das operações de evolução. Ela tem algumas limitações quanto a interface, não oferecendo alguns recursos de interação com os usuários como: navegação entre diferenças, edição de documento e configurações. Desta forma, a ferramenta proposta neste trabalho estende o trabalho de [Perini et al. 2006], oferendo uma melhoria nos recursos de interação, além da possibilidade de executar três algoritmos distintos e comparar os resultados obtidos.

A Tabela 1 apresenta um estudo comparativo entre as diversas ferramentas citadas e o VisualX, que é o objeto deste artigo. O diferencial do VisualX é que, além de dar suporte a diversas funcionalidades encontradas nas demais ferramentas, ele também permite a escolha do algoritmo de *diff* a ser executado.

Tabela 1. Comparação entre ferramentas de visualização de diferenças em XML.

	VisualX	XML Tools	ExamXML	DiffDog	XSDelta
Algoritmos	XyDiff, JXyDiff, XKeyMatch	*	*	*	XyDiff
Distribuição	Acadêmica	Comercial	Comercial	Livre	Acadêmica
Portabilidade	Windows, Linux	Windows	Windows	Windows	Windows, Linux
Navegação entre diferenças	Sim	Sim	Sim	Não	Sim
Edição do documento	Sim	Sim	Sim	Não	Não
Visualização dos casamentos	Não	Não	Não	Sim	Não

* Algoritmo proprietário.

4. A Ferramenta VisualX

O VisualX é uma ferramenta *desktop* desenvolvida para auxiliar os administradores de repositório de dados XML ou usuários interessados nas diferenças entre dois documentos XML a identificar com maior facilidade as modificações efetuadas entre versões distintas de um documento. Para a implementação da ferramenta, foram utilizadas tecnologias de código aberto. Foi utilizada a linguagem de programação Java[1] e, para desenhar e implementar a interface, usou-se a API *Swing*[2], também da plataforma Java. Para acessar, manipular e formatar os documentos XML utilizou-se uma solução, baseada em Java, chamada JDOM[3].

A Figura 1 apresenta a arquitetura do VisualX. A ferramenta recebe como entrada dois arquivos, que são carregados via interface pelo usuário e exibidos em painéis alinhados verticalmente. O documento é apresentado no formato texto, no qual a hierarquia de

[1]http://java.sun.com/
[2]http://java.sun.com/swing
[3]http://www.jdom.org/

aninhamento de elementos é representada por níveis de indentação. Após carregar os documentos nos quais deseja-se aplicar a detecção de diferenças, o usuário pode selecionar um dos três algoritmos disponíveis para fazer essa verificação. Após a detecção o resultado as diferenças são realçadas na tela e o *edit script* pode ser armazenado em formato texto. Na versão atual da ferramenta, as opções de algoritmos são as seguintes:

- **XyDiff** [Cobéna et al. 2002]: Este algoritmo é um dos primeiros algoritmos de detecção de diferenças entre documentos XML propostos na literatura e tem como objetivo realizar a comparação de forma eficiente.
- **JXyDiff** [Potiron 2007]: Este algoritmo para detecção de diferenças entre documentos XML é baseado no XyDiff. Suas principais diferenças em relação ao algoritmo original são a maior portabilidade, pois ele é totalmente escrito em Java e o tamanho reduzido do *edit script*.
- **XKeyMatch** [dos Santos 2006]: Este algoritmo é baseado no XyDiff, porém utiliza chaves XML [Buneman et al. 2001] para realizar o casamento entre os nodos. Com isso, procura adicionar um critério semântico à detecção de diferenças, pois as chaves são capazes de apontar casamentos que, eventualmente, não seriam identificados com a análise baseada apenas na estrutura do documento.

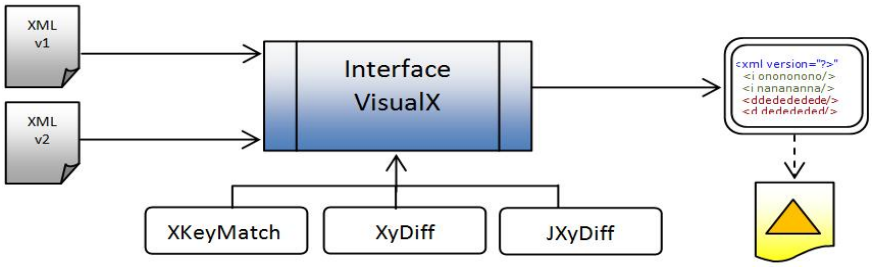

Figura 1. Arquitetura da ferramenta VisualX.

Na Figura 2, uma simplificação do modelo de implementação da ferramenta é apresentado, enfatizando as principais classes que compõe o núcleo de funcionamento da ferramenta. Uma classe abstrata chamada `Algoritmo` foi criada para definir o modelo de comunicação com os programas que implementam os algoritmos de *diff* e fornece a parte genérica das funcionalidades necessárias, que é compartilhada por um grupo de classes derivadas. Cada uma das classes derivadas completa a funcionalidade da classe abstrata adicionando um comportamento específico. Como classes derivadas, são consideradas as classes que implementam os algoritmos suportados, neste caso `XKeyMatch`, `XyDiff` e `JXyDiff`.

Este modelo de implementação permite que a extensão da ferramenta com novos algoritmos seja pontual, ou seja, basta criar uma nova classe derivada da classe `Algoritmo` com os métodos específicos para interpretação do *edit script* deste novo algoritmo. Além disso, todas as interpretações tem como saída uma lista operações (classe `Operacao`) contendo as diferenças encontradas. Dentre os três algoritmos suportados pela ferramenta, apenas o XKeyMatch exigiu a criação de métodos adicionais (`interpretarChaves(k)`). Para os demais, apenas adequou-se os métodos abstratos da classe `Algoritmo`.

Exemplos de *edit scripts* interpretados pela ferramenta são apresentados nas Figuras 3 e 4. A primeira apresenta um exemplo de delta gerado tanto pelo algoritmo

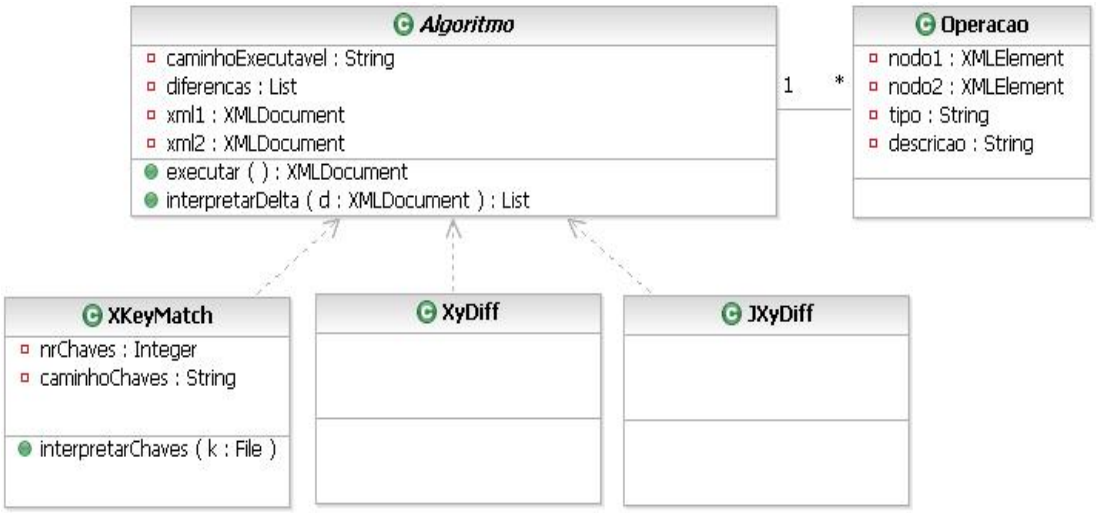

Figura 2. Modelo de implementação da ferramenta.

XKeyMatch como pelo XyDiff. Já a Figura 4 apresenta um exemplo de delta gerado pelo algoritmo JXyDiff. Estas figuras são uma amostra, embora pequena, do quão complexo pode se tornar a análise de um delta quando há um grande número de diferenças detectadas.

```
1. <unit_delta>
2.    <ai a="status" v="10%" xid="4"/>
3.    <au a="concluido" nv="sim" ov="nao" xid="10"/>
4.    <ad a="status" v="80%" xid="10"/>
5.    <d move="yes" par="23" pos="2" xm="(21)"/>
6.    <d par="25" pos="8" xm="(23)"><EsteSemestre/></d>
7.    <i par="25" pos="8" xm="(26)"><SemestreQueVem/></i>
8.    <i move="yes" par="26" pos="2" xm="(21)"/>
9. </unit_delta>
```

Figura 3. Exemplo de *edit script* gerado pelos algoritmos XKeyMatch e XyDiff.

Após realizada a interpretação do delta, as diferenças detectadas pelo algoritmo são mostradas na tela, destacando os elementos XML que sofreram alterações. Esse destaque recebe, por padrão, tons de vermelho para indicar operações de exclusão, verde para indicar operações de inserção, amarelo para indicar operações de alteração e azul no caso de ser uma operação de movimento. Apresentadas as diferenças, o usuário pode navegar por elas, realçando uma a uma. Por fim, o usuário pode decidir salvar ou não o delta gerado.

A Figura 5 ilustra como a visualização de diferenças é apresentada na interface da ferramenta. As versões apresentadas são aquelas utilizadas para gerar os *edit scripts* apresentados nas Figuras 3 e 4. Os elementos excluídos de uma versão para outra são destacados em vermelho na versão V_{t-1} e os inseridos são mostrados em verde na versão V_t. Ainda, alterações em elementos ou dados são enfatizadas na cor amarela em ambas as versões.

Pode ser notado que a interface oferece uma forma muito mais clara de identificar

```
1.<delta>
2.  <DeletedMove move="yes" pos="0:0:3:0"></DeletedMove>
3.  <Deleted pos="0:0:3"></Deleted>
4.  <Inserted pos="0:0:3"></Inserted>
5.  <InsertedMove move="yes" pos="0:0:3:0"></InsertedMove>
6.  <AttributeInserted name="status" value="10%" pos="0:0:0:0"></AttributeInserted>
7.  <AttributeDeleted name="status" pos="0:0:1:0"></AttributeDeleted>
8.  <AttributeUpdated nv="sim" name="concluido" ov="nao" oldpos="0:0:1:0" pos="0:0:1:0"></AttributeUpdated>
9.</delta>
```

Figura 4. Exemplo de *edit script* gerado pelo algoritmo JXyDiff.

as diferenças do que uma análise sobre o delta gerado. Um bom exemplo são as operações de movimentação, onde em uma execução normal do algoritmo de *diff* tais operações são mostradas no *edit script* como uma operação de remoção relacionada com outra de inclusão (Linhas 5 e 8 na Figura 3 e Linhas 2 e 5 na Figura 4), o que torna o trabalho de identificação complexo quando se têm deltas com grande quantidade de operações. Já a ferramenta deste trabalho faz essa interpretação e disponibiliza na interface a operação de movimentação realçando na cor azul, em ambos os documentos, os elementos envolvidos na operação (Linha 13 em ambos os documentos). Neste caso, `aluno4` que era um subelemento de `EsteSemestre` passou a ser um subelemento de `SemestreQueVem`.

Todas as funcionalidades, cores e formas de apresentação disponibilizadas pela ferramenta foram selecionadas através da observação dos melhores recursos disponibilizados por ferramentas semelhantes existentes no mercado e na literatura.

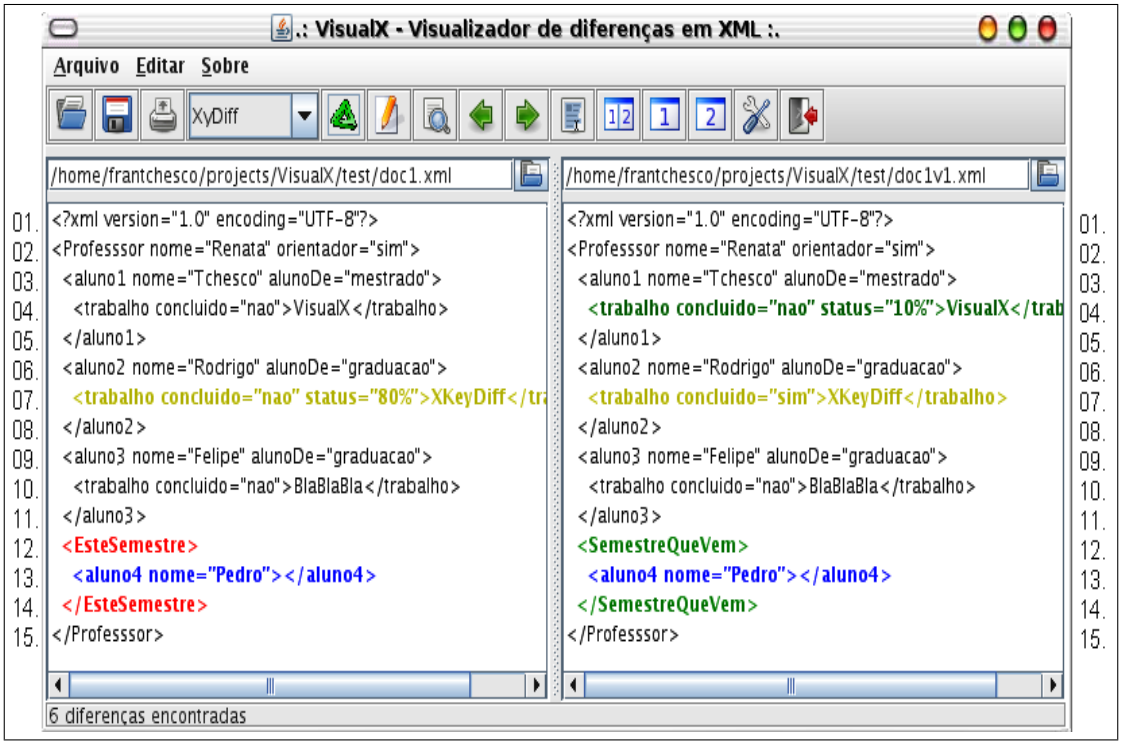

Figura 5. Detecção e apresentação das diferenças na ferramenta VisualX.

Adicionalmente, a ferramenta VisualX oferece recursos de interação com usuário como: edição dos documentos abertos, o que permite realizar ajustes nos documentos sem a necessidade de abri-los em um outro editor; configurações de cores; teclas de atalhos, que são úteis para usuários com preferência por utilizar o teclado; e dicas de contexto, que oferecem informações sobre os recursos da ferramenta.

5. Conclusão

O objetivo deste trabalho é apresentar uma ferramenta que, através de uma interface gráfica, facilite a análise de diferenças entre duas versões de um documento XML, detectadas por um algoritmo de *diff*. Para isso, foi realizado um estudo de alguns algoritmos de detecção de diferenças em documentos XML que geram o resultado em formato texto. Também foram analisadas ferramentas existentes que oferecem uma interface gráfica para a visualização das diferenças. Isto serviu como parâmetro para que a interface desenvolvida nesse trabalho pudesse reunir os recursos considerados relevantes para proporcionar uma boa interação com o usuário. A ferramenta implementa de forma simples os recursos necessários para disponibilizar ao usuário uma interface única que permite identificar de forma clara as diferenças encontradas por três algoritmos de *diff* distintos. Entre esses algoritmos, merece destaque o suporte ao XKeyMatch, pois este permite a definição de chaves como um aditivo semântico ao processo de comparação de diferenças.

Para as próximas versões da ferramenta, além da integração de outros algoritmos de *diff*, podem ser indicadas pelo menos três funcionalidades desejáveis. A primeira funcionalidade é a apresentação visual dos casamentos realizados pelo algoritmo de detecção. Esse recurso permitiria ao usuário identificar possíveis erros na fase de casamento dos elementos do documento. A segunda melhoria consiste em permitir ao usuário fazer correções nos casamentos identificados e com isso realizar a detecção de diferenças de forma incremental. Como terceira funcionalidade, seria interessante que o *edit script* armazenado pela ferramenta utilizasse um formato XML padrão ao invés daquele gerado por cada algoritmo. Dessa forma, a própria ferramenta VisualX poderia ser utilizada para comparar os resultados dos diversos algoritmos de *diff* e refinar ainda mais a identificação de diferenças.

Agradecimentos: Agradecemos Augusto Belotto Perini por disponibilizar o código-fonte da ferramenta XSDelta, que foi utilizada como modelo para o desenvolvimento do VisualX.

Referências

[A7Soft 2007] A7Soft (2007). ExamXML. Disponível em http://www.a7soft.com/examxml.html.

[Altova 2007] Altova (2007). DiffDog - XML - Aware diff/merge tool for file and directory differencing. Disponível em http://www.altova.com/products/diffdog/diff_merge_tool.html.

[Buneman et al. 2001] Buneman, P., Davidson, S., Fan, W., Hara, C. S., and Tan, W.-C. (2001). Keys for XML. In *Proceedings of the 10th International World Wide Web Conference (WWW'10)*, pages 201–210, Hong Kong.

[Cobéna et al. 2002] Cobéna, G., Abiteboul, S., and Marian, A. (2002). Detecting changes in xml documents. In *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*, pages 41–52, Washington, DC, USA.

[dos Santos 2006] dos Santos, R. C. (2006). XKeyMatch: Um algoritmo de detecção de diferenças entre documentos XML. Tese de mestrado, UFPR, Curitiba - Brasil.

[Perini et al. 2006] Perini, A. B., da Silveira, V. N. K., and de Matos Galante, R. (2006). XSDelta: Uma ferramenta visual para comparação de esquemas XML. In *III Sessão de Demos (SBBD´06)*, pages 7–12, Florianópolis, SC, Brasil.

[Studio 2007] Studio, S. (2007). XML Differencing. Disponível em http://www.stylusstudio.com/xml_differencing.html.

[Potiron 2007] Potiron, P. (2007). JXyDiff. Disponível em http://potiron.loria.fr/projects/jxydiff.