XESB (XML Editor Schema Based)
Um editor para as massas

Ricardo Queirós
c0165013@alunos.dcc.fc.up.pt

José Paulo Leal
zp@ncc.up.pt

Referência XESB:
http://www.dcc.online.pt/~c0165013/

Abstract. Este documento descreve a especificação e implementação duma aplicação Web para edição assistida de documentos XML chamada XESB. O objectivo principal é a criação de um editor destinado a utilizadores sem conhecimentos de XML. A aplicação utiliza técnicas de edição estrutural para garantir a validade do documento durante o processo de edição. A definição de tipos de documentos é feita em XML Schema. Como motivação, este editor poderá vir ser integrado em sistemas Web que requeiram a configuração de documentos em linguagem XML, ou mais genericamente, poderá servir como um serviço interactivo de produção de documentos XML.

1. Introdução

Os documentos XML foram pensados para serem processados automaticamente e poderem também ser lidos por pessoas, mas não foram pensados para poderem ser editados. As regras de boa-formação tornam a interpretação dos documentos não ambígua mas exigem maior rigor por parte de quem os produz. A especificação XML determina regras rígidas para que os documentos possam ser considerados bem-formados que são difíceis de manter por um utilizador humano. Esse rigor pode facilmente ser obtido se os documentos forem gerados por um programa mas exige muita disciplina a uma pessoa que produza um documento XML num editor de texto genérico. Por outro lado as anotações XML tornam os documentos pouco sucintos o que, não sendo um problema para um computador, exige muito mais esforço para uma pessoa. Um documento XML é consideravelmente mais extenso do que um documento escrito numa linguagem convencional [2], já que as anotações devem ser auto-explicativas. Esta verbosidade está associada aos objectivos iniciais do desenho do XML, nomeadamente a facilidade de leitura dos documentos XML por um humano e a não necessidade de concisão das anotações [3]. Existe pois, uma

dificuldade intrinseca à edição de documentos XML, devido quer ao rigor da sua sintaxe quer à verbosidade das suas anotações.

Observando os vários cenários de necessidade de autoria de documentos XML, bem como a dificuldade intrínseca do seu processo de edição, pode-se concluir que é importante existir editores que produzam documentos XML válidos. Actualmente existem vários editores que permitem a criação e edição de documentos XML, desde editores de texto genéricos que extendem as suas capacidades de edição através de *plugins* e que permitem integrar *parsers* que irão conduzir o processo de edição, a editores específicos de XML. A generalidade destes editores exige o conhecimento da norma XML, limitando assim o leque de possíveis utilizadores do editor, e obrigam a uma prévia instalação de software inibindo assim o seu uso em situações de transição onde apenas se quer criar um número limitado de documentos.

Perante este cenário partiu-se para a criação de um novo tipo de editor chamado XESB – XML Editor Schema Based, que se baseia numa aplicação Web (acessível via navegador) e que não requer o conhecimento de qualquer formalismo. O editor utiliza definições de tipo de documentos que guiam todo o processo de edição garantindo que os documentos criados são sempre válidos.

1.1 Requisitos de desenho

Tendo por objectivo o desenvolvimento de um sistema de edição de documentos XML destinado a utilizadores sem conhecimentos do formalismo, foram identificados os seguintes requisitos de desenho:

- **Abstracção do conhecimento da sintaxe:** Abstrair o utilizador do conhecimento da sintaxe através de interfaces WYSIWYG; esta abordagem alarga o leque de potenciais utilizadores, fazendo com que os mesmos não se percam com os detalhes sintácticos das tecnologias inerentes, concentrando-se apenas no seu conteúdo;
- **Validação:** Guiar o processo de edição de um documento através duma definição formal da linguagem, obtendo assim, por construção, documentos válidos;
- **Operações genéricas sobre fragmentos de documentos XML:** Providenciar operações genéricas (copiar, mover, inserir, remover, clonar) sobre fragmentos de documentos XML flexiblizando assim o processo de edição de um documento e promover também operações entre documentos;
- **Procura:** Disponibilizar mecanismos de procura sobre o texto do documento e que beneficiem da natureza estrutural do mesmo;
- **Modelos:** Usar definições de tipo de documento (DTD) ou documentos esquemáticos (XML Schema) como modelos para instanciar documentos. Esta instanciação é automática e configurável. Os documentos gerados irão herdar as características da "classe" à qual foram instanciados;
- **Inferência de esquemas:** Inferir esquemas a partir de documentos instâncias;
- **Transformação:** Gerar interfaces familiares para edição de documentos XML através da transformação dos documentos usando folhas de estilo XSL. Por omissão deverão existir stylesheets genéricas;

- **Vistas:** Suportar várias perspectivas dum documento XML;
- **Exportação:** Providenciar mecanismos de exportações dum documento para vários formatos (PDF, PS, TXT, SVG, etc.).

1.2 Abordagem: aplicação Web e edição estrutural

A Web é um meio universal para partilha e publicação de documentos. Partindo dessa característica de ubiquidade, e derivado dos requisitos de desenho referidos anteriormente, foi desenvolvida uma aplicação Web para edição assistida de documentos XML. A aplicação utiliza técnicas de edição estrutural para garantir a validade do documento durante o processo de edição. A definição de tipos de documentos é feita em XML Schema, podendo-se utilizar DTD's que são convertidos para a linguagem de esquemas definidas pelo W3C. O suporte de DTD's é justificado pela sua preponderância nas aplicações XML [4]. Estes formalismos definem a estrutura e o tipo de conteúdo que os documentos a ele instanciados poderão ter. Este editor poderá vir ser integrado em sistemas Web que requeiram a configuração de documentos em linguagem XML, mais genericamente poderá servir como um serviço interactivo de produção de documentos XML. Seguidamente justificam-se as principais características do editor, nomeadamente, ser uma aplicação web e usar edição estrutural. Os modelos a considerar eram:

- Uma aplicação tradicional instalada no computador do utilizador;
- Uma aplicação baseada num navegador web, usando as suas possibilidades de processamento de documentos e a web apenas para descarregar o código da aplicação;
- Uma aplicação cliente/servidor em que o processamento do documento é mantido do lado do servidor e apenas a visualização e interacção do lado do cliente.

No caso de uma aplicação autónoma é difícil, converter o XML para um formato visual (XHTML) onde fosse possível interagir com o documento. Converter o documento XML em XHTML é a abordagem mais imediata, mas as bibliotecas que suportam a navegação sobre documentos em aplicações autónomas, como a classe JeditorPane do pacote Swing suportam apenas versões antigas do HTML e não suportam JavaScript.

Uma aplicação assente integralmente no navegador iria cingir a portabilidade da aplicação a navegadores com forte suporte para as tecnologias DOM, XML, XSL e JavaScript. Um exemplo interessante desta abordagem, é o editor BitFlux XML Editor [5] , que permite a edição de documentos XML e que é escrito em JavaScript. Este editor está limitado ao navegador Mozilla.

Adoptando o modelo cliente-servidor beneficiámos simultaneamente da capacidade de navegação e interacção do XHTML nos navegadores e da maior capacidade para processamento do lado do servidor. Para esse processamento é utilizado a linguagem JAVA devido ao seu grande suporte para XML.

No XESB o utilizador deve apenas executar operações sobre o documento compatíveis com o documento esquemático associado. Esta ideia é o conceito base da edição estrutural [6] ou edição dirigida pela sintaxe. A ideia central da edição estrutural é condicionar a edição dum texto numa linguagem mediante uma interface gráfica que permita apenas expansões sinctaticamente válidas. Este processo é designado de compilação incremental. Esta baseia-se na existência duma definição formalda linguagem que permite determinar se um texto lhe pertence ou não. Nos editores estruturais clássicos essa definição formal é uma gramática, em geral livre de contexto. Para o caso do XML a definição formal de linguagem é, por exemplo, um DTD ou um XML Schema, complementando as regras básicas de boa formação do XML. Um editor estrutural é um editor de texto especializado para uma dada linguagem e dirigido pela sintaxe da mesma. Quer isso dizer que o editor vai conduzindo o utilizador de acordo com a gramática da linguagem, preenchendo automaticamente quaisquer anotações ou valores fixos de elementos e atributos que se encontrem definidos no esquema e podendo indicar as alternativas disponíveis em cada momento. Assim, o utilizador não se perde com detalhes sintácticos do documento, já que o editor só admite uma estrutura correcta. podendo concentrar-se no conteúdo (semântica) do que está a escrever [7].

A edição estrutural faz mais sentido em documentos XML do que em linguagens de programação. porque nestas últimas a gramática é sucinta e é normalmente bem conhecida por quem a usa. Por isso a edição estrutural nunca chegou a ser muito usada nos ambientes de programação para os quais foi concebida pois torna o processo de edição de programas mais complexo e moroso. Nos documentos XML, o cenário é diferente. O XML têm regras de boa-formação rigorosas, mas os documentos não tem de ser sucintos. Um documento XML é consideravelmente mais extenso, já que as anotações devem ser auto-explicativas, exigindo muito mais esforço de edição se forem escritos num editor de texto convencional. Existindo um DTD ou Schema associado ao documento XML é vantajoso usar edição estrutural pelas seguintes razões:

- Possibilitam a validação estrutural do documento durante a edição;
- Disponibilizam ajuda contextual ao utilizador, prevenindo-o sempre que este incorrer em erro;
- Fornecem a lista de anotações válidas num determinado ponto do documento, evitando a memorização da sintaxe da linguagem a editar;
- Reconhecem e realçam no texto, as anotações de início e fecho dos elementos, os atributos e as entidades, definidos por um DTD ou Schema[1].

[1] Podia ser feito recorrendo apenas a expressões regulares

2. Estado de Arte

O XML é um formalismo para documentos de texto anotados. Por esse facto é possível editar um documento XML utilizando um editor de texto, como por exemplo o Notepad. Estes editores possuem apenas operações genéricas de edição, como a inserção, remoção e substituição de caracteres e a sua selecção. Nestes casos a edição de XML é morosa e é necessário conhecer bem o formalismo, já que este tipo de editores não fornece qualquer ajuda em termos da sintaxe do XML, nem possui qualquer tipo de suporte para a integração de DTD's ou XML Schema com vista à validação do documento. A difusão do XML levou a que alguns editores de texto começassem a suportar XML adaptando algumas funcionalidades a este género de documentos. Estas funcionalidades estão dependentes da linguagem a editar. No contexto do XML, salientam-se o realce das anotações usando cor, a capacidade de expandir e colapsar elementos, a indentação, a criação automática de anotações de fecho e navegação entre anotações, a inserção de entidades, entre outras. Esta adaptação surge, muita vezes, na forma de ***plugins*** que extendem as capacidades do editor de texto genérico. Um exemplo é o editor Emacs através do ***plugin*** PSGML [8] que permite uma edição de documentos XML mais rápida e com um melhor suporte.
Alguns editores de texto genéricos utilizam expressões regulares para dotar o editor de características importantes na edição de documentos duma linguagem, e podem ser facilmente extendidos para XML. Este mecanismo, no entanto, não resolve questões relacionadas com a boa-formação ou validação de um documento XML. Para essa verificação, é necessário extender estes editores com módulos/***plugins*** ou simplesmente aplicações externas, que irão dotar os mesmos, com capacidades extra para a edição assistida de documentos XML. Se o documento XML tiver associada uma especificação da estrutura (DTD ou XML Schema), é necessário haver uma validação estrutural do documento, sendo esta assegurada pelo ***parser*** validador que está associado ao editor. Nestes editores de texto genéricos e extensíveis, destacam-se o Emacs e o jEdit.
Actualmente, existem já editores específicos para linguagens XML. Esta nova abordagem oferece um conjunto de funcionalidades específicas que permitem uma edição mais rápida e consistente dos documentos.Contudo ao utilizar estes editores, perdem-se as capacidades genéricas dos editores de texto. Eis uma enumeração de algumas das características principais existentes nos editores XML actuais:

- Suporte a transformação de documentos XML(XSLT/XSL-FO) e depuramento na sua execução;
- Suporte DTD, XML Schema, RELAX-NG, etc;
- Integração de ferramentas de controlo de versões e de deployment (CVS e Ant);
- Importação e exportação para BD's relacionais;
- Múltiplas visões do documento;
- Fácil edição de documentos (controlos avançados, ambiente gráfico, etc.)

Neste tipo de editores destacam-se o Authentic da Altova, o xMetal da SoftQuad e o BitFlux da Oscom.

3. Tecnologias usadas no XESB

Na implementação do XESB optou-se por usar standards, API's e pacotes que implementam essas API's, bem como, ferramentas "open-source" e multi-plataforma. Garante-se assim uma maior portabilidade da aplicação.

Foi adoptado o modelo cliente-servidor, em que o cliente é um navegador Web (como o Mozilla, IE, etc) e o servidor é um contentor de servlets/JSP (como o Tomcat, Resin, etc.). A lógica de negócio fica a cargo dos JavaBeans e a camada de apresentação através da mescla das especificações JSP / XHTML apoiadas pela validação JavaScript e a formatação CSS. A escolha do JAVA como linguagem de programação do lado do servidor deve-se à característica de portabilidade intrínseca à linguagem e sobretudo ao facto de haver um bom suporte para XML no Java. O desenvolvimento do trabalho foi feito no ambiente Eclipse. Para o processamento dos documentos XML e a sua associação com uma especificação da estrutura (DTD ou XML Schema) utilizaram-se as seguintes tecnologias:

- Linguagens de transformação e formatação: XSL (XSLT, XPATH e XSL-FO);
- Interfaces aplicacionais Java a documentos XML: JAXP (DOM/SAX, TRAX);
- Pacotes de Processamento de documentos XML: Xerces (Parser XML) , Xalan (Processador XSL) e NekoDTD (conversão DTD para XML Schema).

Em termos de navegadores o suporte é garantido aos navegadores Mozilla (FireFox) a partir da versão 0.8 e Internet Explorer a partir da versão 5.0, o que representa quase 95% dos navegadores em utilização [9].

4. XESB

O XML é uma especificação de documentos anotados que permite definir uma linguagem usando DTD ("Document Type Definition") ou o Schema. Estas definições correspondem à especificação da gramática se quisermos criar uma analogia [10] com as linguagens de programação. O XESB permite a edição de documentos XML tipificados e apesar de suportar as duas variantes – DTD e XML Schema – este último é a base para a compilação incremental. Esta opção deve-se ao facto de uma definição de tipo de documento XML Schema ser em si um documento XML (reutilização de API's para acesso e manipulação); ao suporte para diferentes tipos de dados; ao suporte de namespaces e às possibilidades de desenvolvimento modular que estes têm face aos DTDs.

O XESB é constituído por vários componentes que partilham um repositório de dados comuns contendo informações sobre um conjunto linguagens e documentos associados. A Fig. 1 ilustra uma visão funcional dos módulos constituintes do XESB.

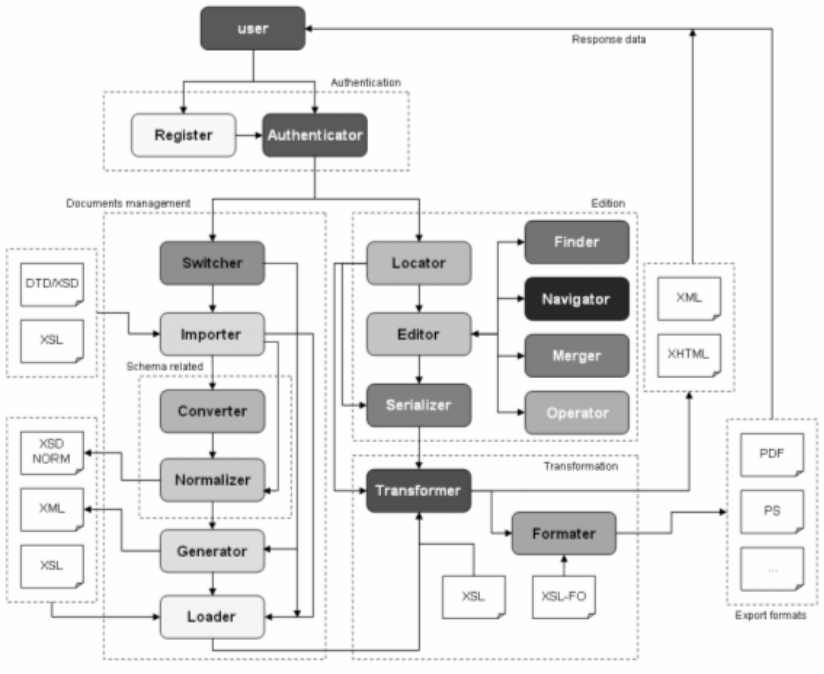

Fig. 1. Arquitectura funcional do sistema XESB

A arquitectura funcional do XESB foca quatro grandes áreas:

- **Autenticação**: Filtra o uso da aplicação a utilizadores com registo, disponibilizando uma área de trabalho no servidor (workspace);

- **Gestão de documentos:** Controla todo o processamento dos documentos desde o carregamento para memória dos documentos XML até à criação de um documento instância. Trata também da normalização do documento esquemático associado e da conversão de DTD's para XML Schema;

- **Edição:** Controla o processo de edição, mapeando todas as acções do utilizador. Localiza o elemento a editar, combina os vários fragmentos dos documentos para resposta ao utilizador. Controla também o processo de procura e navegação no documento, bem como a serialização dos documentos em memória.

- **Transformação:** Define as transformações a serem aplicadas às respostas preparadas pela área de edição. A linguagem objectivo das transformações é o XHTML. Inclui opções para a exportação do documentos para outros formatos (PDF, PS, etc.).

O utilizador do protótipo deverá fazer um registo prévio criando assim uma área de trabalho, constituida por um espaço físico em disco (servidor) para armazenamento dos documentos gerados. Se já possuir registo, é necessária apenas a entrada dos dados de autenticação. Para a criação de documentos instância deverá associar um tipo de documento (existem 3 pré-definidos) e um "template" (se não existir nenhum para determinado tipo é carregado um por omissão). Após estas associações é gerado um esquema normalizado (baseado no tipo de documento seleccionado) e um novo documento instância apenas com a estrutura e pronto para ser editado. No caso de se querer criar um novo tipo de documento é necessária a sua importação prévia para o sistema XESB. Aceitam-se dois formalismos: DTD (sofre uma conversão para XML Schema) ou XML Schema. O processo restante é igual ao referido anteriormente.

Após a selecção de um XML Schema e de uma stylesheet que irá definir uma interface para a edição, o utilizador ao interagir com elementos de formulário vai produzir documentos XML que estão de acordo com a definição formal associada. Esta abordagem permite assim abstrair o utilizador dos detalhes sinctáticos do documento, podendo concentrar-se no seu conteúdo

Após a geração do documento instância, este é extendido por um conjunto de atributos associado a um namespace do sistema XESB: http://www.alunos.dcc.fc.up.pt:8080/xesb/ . Este conjunto tem como missão o controlo de pedidos e respostas que resultam da interacção do utilizador com o editor.

```
<nomeElemento  xesb:id | xesb:gid | xesb:find |
xesb:select | xesb:*>...</nomeElemento>
```

O atributo id identifica um elemento editável. A Fig. 2 demonstra todo o fluxo de informação que é executado aquando da interacção com o utilizador. Após a selecção de um elemento para edição é enviado para o servidor o identificador do elemento a ser editado (1).

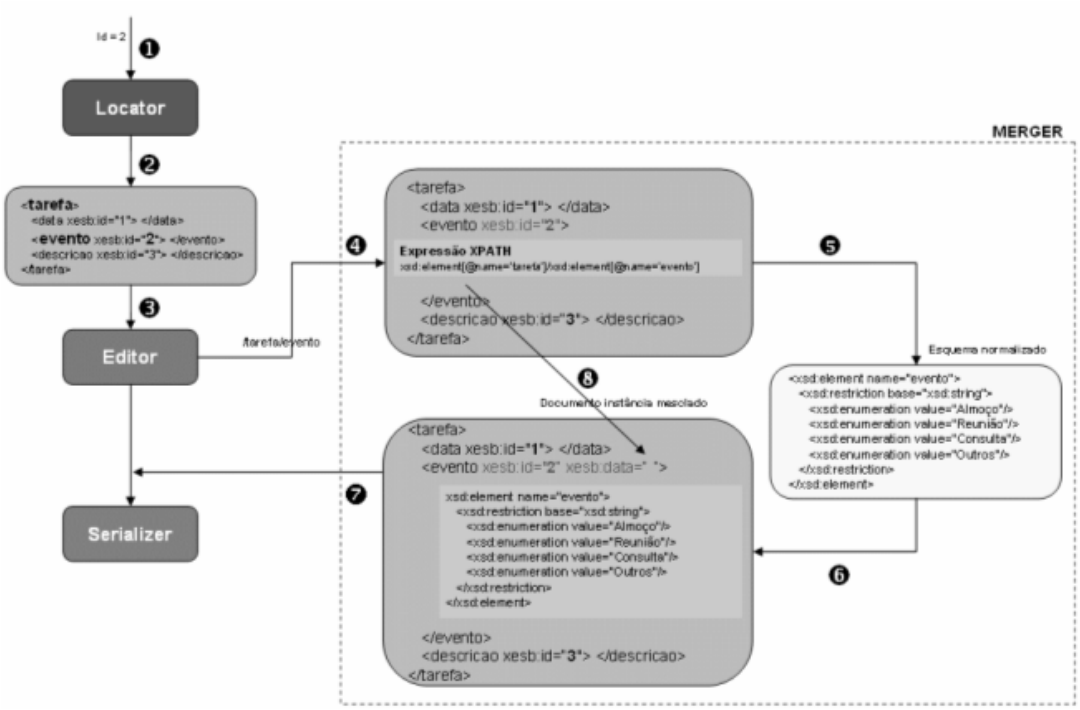

Fig. 2. Pedido de edição de um elemento

O módulo Locator localiza (2) o elemento através do identificador (atributo id) e constroi o caminho absoluto do elemento a editar passando a informação para o Editor (3) que, por sua vez, verifica que se trata de um pedido de edição de um elemento. O Editor precisa de saber que restrições esse elemento tem na definição do esquema e envia para o componente Merger o caminho absoluto do elemento a editar na árvore do documento instância (4). O Merger recebe a informação, mapeia o caminho para uma expressão Xpath que aponta para a declaração do elemento no esquema (5). Note-se que como o esquema está normalizado, sabe-se que toda a informação respeitante ao elemento encontra-se agregada a ele como um nó filho. Não é pois necessário estar atento a referências e apontadores pois toda a informação está compactada no mesmo local (a principal razão da normalização do esquema). De seguida, faz-se a fusão do fragmento resultante da execuçaão da expressao Xpath com a árvore do documento (6). Após a fusão o Merger redirecciona o fluxo para o Serializer (7) que grava as alterações que serão posteriormente transformadas via Transformer. De realçar que se o elemento a editar já contiver algum valor este é adicionado no documento resultado como valor do atributo xesb:data (8). Desta maneira consegue-se passar o valor actual, de novo para o utilizador.

O Transformer controla a transformação do documento mesclado (XML + XSD). A Fig. 3 retrata os templates utilizados neste processo. A parte do documento que diz

respeito ao XML fica a cargo do template carregado em memória e que é específico ao tipo de documento (se não se tiver associado nenhum é utilizado um por omissão – xesbDefault.xsl), enquanto que qualquer excerto XSD é tratado pelos ficheiros de transformação xesbEdit.xsl e xesbFunc.xsl, ou seja, quaisquer elementos/atributos do namespace do XML Schema são transformados por estes templates que têm por missão a transformação de declarações de elementos (e respectivas restrições) em elementos de formulário associados com módulos pré-definidos e codificados em JavaScript para validação do lado cliente.

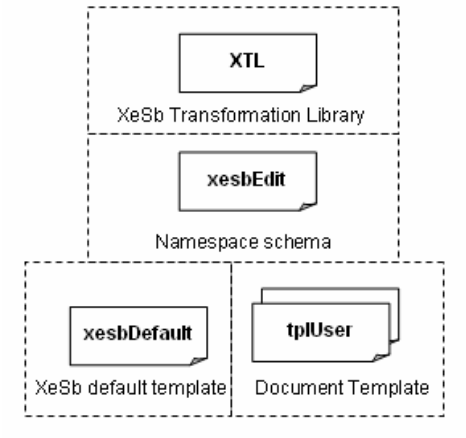

Fig. 3. Templates do componente Transformer

A linguagem objectivo das transformações é XHTML que é seguidamente consumido pelo navegador Web usado pelo utilizador. Este componente utiliza o módulo Formatter, no caso da necessidade de exportação dos dados para formatos diferentes, tais como, PDF, PS, TXT, AWT, SVG, MIF ou PCL. Neste caso a linguagem objectivo da transformação (usa-se o ficheiro xesbDefaultFO.xsl) é o XSL-FO e, juntamente com o documento instância XML, servem de input para um formatador[2] que transforma os dados para o output desejado. De salientar ainda neste componente Transformer a existência de uma biblioteca de funções (XTL - XESB Transformation Library), que permite a criação de templates específicos para novos tipos de documentos. O ficheiro editado é mantido na área de trabalho para posterior acesso ou mesmo o seu descarregamento para o computador do utilizador.
Em termos de visualização do documento, o XESB disponibiliza o modo de edição onde o utilizador interage com formulários, o modo em árvore onde o écran é sub-dividido em 2 frames (Fig.4): árvore do tipo de documento e o formulário de edição e uma visualização do código do documento instância.

[2] Usou-se o open-source da Apache chamado FOP e que é uma implementação parcial do standard XSL-FO 1.0 do W3C

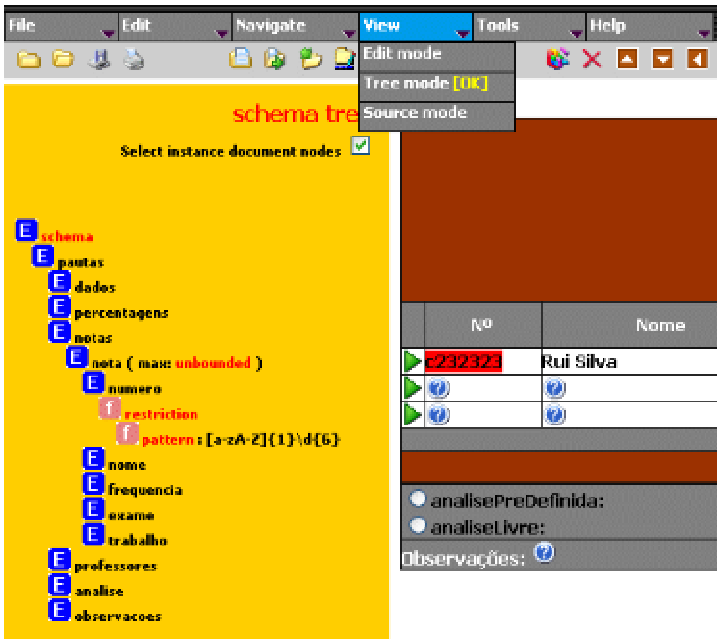

Fig. 4. Visualização em árvore do tipo de documento a editar

5. Conclusão

O XESB é um protótipo de um editor de XML que serviu essencialmente para validar uma abordagem: a utilização de edição estrutural e estruturação como aplicação web cliente-servidor. Embora esta abordagem tenha sido claramente validada pelo XESB este protótipo revelou algumas deficiências que esperamos colmatar em projectos futuros nesta área.

- **Suporte da especificação XML Schema:** O XESB não tem suporte a algumas áreas constituintes da especificação XML Schema, nomeadamente, o modelo de conteúdo de opção (elementos choice como nós internos), as definições de restricões de identidade (unique, key, keyref, selector e field), as notações (notation), a gestão de documentos (redefine, import, include) e uma característica da declaração de elementos (substitutionGroup). O facto de não aceitar modelos de conteúdo de opção em nós internos e o fraco suporte para conteúdos mistos, são talvez a maior limitação deste editor. Estas limitações devem-se essencialmente ao esforço de desenvolvimento necessário face ao tempo disponível para a sua implementação. O suporte a outros formalismos esquemáticos (RELAX NG, Schematron, etc.) é também desejável.

- **Edição de documentos esquemáticos e de transformação:** O XESB permite a edição de documentos XML, mas os documentos esquemáticos e de transformação são também documentos XML, podendo-se assim beneficiar e utilizar as capacidades do próprio XESB. Mas devido às limitações apresentadas anteriormente e à estrutura geral do sistema, essa funcionalidade, que se considera primordial, não está incluída;

- **Inferência de documentos esquemáticos através de vários documentos instância:** A existência dum processo de inferência de um documento esquemático deverá ter como base vários documentos instância XML, garantindo assim uma melhor caracterização do mesmo. O TRANG um software de conversão da Thai Open Source Software Center Ltd poderá vir a ser integrado. A ideia principal é que através de um (ou vários) documentos XML se possa editar a própria linguagem definida no documento inferido, para tal é necessária a inclusão do Schema do próprio documento esquemático;

- **Interface gráfico para procura XPath:** A procura através de expressões XPath é uma das mais importantes funcionalidades de um editor XML. Com vista a alargar este mecanismo para utilizadores leigos nesta tecnologia era interessante incluir uma interface gráfica que permitisse a manipulação das expressões abstraindo o utilizador do formalismo inerente.

Em suma, existem várias funcionalidades a integrar, em trabalho futuro, tendo em vista a evolução do editor XESB, com vista a assumir-se como um ambiente integrado para o desenvolvimento de aplicações XML.

Referências

[1] W3C World Wide Web Consortium (http://www.w3.org).
[2] JavaML: A Markup Language for Java Source Code, Greg J. Badros - Dept. of Computer Science and Engineering - University of Washington - Seattle, WA USA
[3] Professional XML, Mark Birbeck and others, Wrox, 2ª edição, 2001.
[4] The XML Web: a First Study - Laurent Mignet, Denilson Barbosa, Pierangelo Veltri - 2003.
[5] BitFlux XML Editor - http://bitfluxeditor.org/
[6] The Synthesizer Generator: A System for Constructing Language Based Editors. Texts and Monographs in Computer Science. Thomas Reps and Tim Teitelbaum Springer Verlag, 1989.
[7] INES - Ambiente para Construção Assistida de Editores Estruturados baseados em SGML, Alda Reis Lopes, José Carlos Ramalho, Pedro Henriques - Departamento de Informática - Universidade do Minho – 1997
[8] Psgml: a gnu emacs major mode for editing sgml, Lennart Staflin.
[9] W3Schools (http://www.w3schools.com).
[10] Anotação estrutural de documentos e sua semântica, José Carlos Leite Ramalho, PhD thesis, Escola de Engenharia - Departamento de Informática - Universidade do Minho, 2000.