Projeto LexML Brasil – Inteiro Teor

XML Schema para Normas, Julgados e Projetos de Norma

Versão 0.6 (maio/2008)

João Alberto de Oliveira Lima (Senado/PRODASEN – joaolima@senado.gov.br)

Fernando Ciciliati (Senado/Interlegis – fernandociciliati@interlegis.gov.br)

1. Introdução

Este documento apresenta princípios e conceitos utilizados na definição do esquema XML para estruturação dos textos de normas, julgados e projetos de normas do Brasil do projeto LexML Brasil. O *schema* "lexml06.xsd" é fortemente baseado nos esquemas dos projetos NIR (*Norme in Rete*) e AKOMA NTOSO (ambos editados pelo Prof. Fabio Vitali da Universidade de Bologna). Neste documento será dada ênfase aos detalhes do ordenamento jurídico brasileiro e da nossa técnica legislativa. Em algumas seções, haverá a adaptação e/ou tradução de alguns trechos do documento de referência do projeto AKOMA NTOSO ("Akoma Ntoso 1.0 - Release Notes" – versão 1.0 release 22/10/2007).

Na época da implantação do projeto NIR, a linguagem *XML Schema* ainda estava em processo de definição. Atualmente, o esquema do projeto NIR está disponível tanto na notação DTD quanto em *XML Schema*. Como a linguagem *XML Schema* possui mais recursos de validação do que a DTD, decidiu-se por utilizar apenas esta nova modalidade para modelagem dos documentos textuais no projeto LexML Brasil.

A leitura deste documento requer o conhecimento básico de XML bem como de *XML Schema* que viabiliza a definição das regras de modelagem dos documentos textuais.

2. Esquemas Rígido e Flexível

Devido à falta de padronização da técnica legislativa em relação às três esferas e à existência de normas federais com articulação incompatível com a Lei Complementar nº 95 (principalmente normas anteriores a 1998), decidiu-se pela criação de dois esquemas, a exemplo da estratégia do projeto NIR, conforme abaixo:

- Esquema rígido – Normas e projetos de normas que seguem as regras da LCP-95.
- Esquema flexível – Por conter menos restrições de combinação entre os elementos, permite mais combinações na articulação dos dispositivos. Se aplica a normas e projeto de normas que não seguem a LCP-95. Serve também como um denominador comum entre os esquemas rígidos definidos para cada ordenamento jurídico.

Todo documento válido no LexML rígido é também válido no LexML flexível. No entanto, o inverso não é verdadeiro. Estados e municípios que não utilizam as regras de estruturação conforme a LCP-95, deve definir seu próprio esquema rígido. Normas federais anteriores à LCP-95 que não consigam ser validadas no esquema rígido, devem utilizar o esquema flexível.

3. Namespace

As instâncias de documentos LexML são completamente qualificadas, isto é, utiliza-se o *namespace* "http://www.lexml.gov.br/0.6" como *namespace default* para todos os elementos.

"Apesar de alguns elementos utilizarem o mesmo nome de elementos HTML, e de fato terem sido importados diretamente do vocabulário HTML, para simplificar, decidiu-se utilizar apenas um *namespace*, permitindo que todos os elementos sejam qualificados de forma idêntica. O resultado disto é que é possível especificar o *namespace LexML* como *default* e não utilizar prefixos no documento da instância XML, ao mesmo tempo que se mantém a qualificação completa dos elementos." (adaptado de F. Vitali, 2006)

O LexML-br utiliza três atributos que devem ser codificados com o namespace explícito, conforme a seguir: "xml:*lang*" e "xml:*base*" do *namespace* `http://www.w3.org/XML/1998/namespace;` e "xlink:href" do namespace `http://www.w3.org/1999/xlink.`

O atributo *lang* deve ser utilizado no elemento *inline* `<span>` para delimitar expressões escritas em outras línguas, tais como, latim (la), inglês (en), espanhol (es), alemão (de), utilizando o código de acordo com o padrão ISO 639-1 (2002).

O atributo *base* deve ser utilizado no elemento `<Alteracao>` ou `<RemissaoMultipla>` para identificar a URN da norma que está sendo alterada ou referenciada. Todos os atributos "*href*" que fazem parte dos elementos contidos no elemento `<Alteracao>` são relativos à URN indicada no atributo "base".

4. Nome e Conteúdo de Elementos e Atributos

Uma das principais características de um documento XML é a sua capacidade de se auto-descrever, isto é, cada segmento do conteúdo textual é delimitado por marcações (*tags*) que possuem nomes significativos.

Todos os elementos do LexML (exceto os "importados" de outros vocabulários – ex. HTML) utilizam a notação *UpperCamelCase*. Os atributos utilizam a notação *lowerCamelCase*.

Ao se definir uma nova linguagem XML para uma determinada comunidade é comum surgirem dúvidas quanto à escolha entre o construtor elemento ou atributo em uma determinada situação. Utilizou-se a seguinte diretriz no LexML: o conteúdo textual dos elementos deve ser suficiente para reproduzir o texto constante no documento em oficial (em papel ou arquivo digitalmente assinado) na ordem em que ele se apresenta; aos atributos está reservado o papel de informações de controle do conteúdo texto, tais como identificadores, referências, indicação da língua etc. No caso de acórdãos, onde há repetição do cabeçalho em algumas de suas páginas, decidiu-se codificar estas informações em um elemento `<CabecalhoAcordao>` separado dos outros componentes textuais do acórdão (relatório, voto etc) com o objetivo de minimizar a redundância de informações.

Não são utilizados diacríticos nem a letra “ç” nos nomes de elementos, atributos ou no conteúdo de atributos “id”. Por serem construtores voltados principalmente para o processamento técnico do documento, faz parte da prática da informática omitir estes sinais que nem sempre estão disponíveis nos teclados.

4.1. *Encoding* e UTF-8

As instâncias de documentos da linguagem LexML devem utilizar o *encoding* UTF-8. Esta é a codificação *default* para o XML e foi escolhida porque oferece o armazenamento em um único byte para os caracteres mais comuns e permite a utilização dos caracteres definidos no padrão Unicode. A codificação ISO-8859-1 (muito difundida no Brasil) sempre utiliza um único byte por caractere, e, por isso é bem mais limitada que a UTF-8. Por exemplo, esta codificação não possui as aspas angulares: “ (abre aspas) e ” (fecha aspas). Seria necessário utilizar entidades ou referências numéricas para codificá-las em um documento XML caso fosse utilizado a codificação ISO-8859-1.

Outra vantagem da UTF-8 é a possibilidade de armazenamento de documentos textuais de outros países que utilizam um conjunto de caracteres não latino.

4.2. Cuidados com alguns caracteres

É muito comum encontrar na Internet textos de normas codificados com caracteres errados. Um dos erros mais comuns é o símbolo de ordinal “ º ” (código 186), utilizado na identificação de normas, como por exemplo: “Lei nº 8.112 ...” ou “§ 2º do art. 6º”. Este símbolo algumas vezes é codificado como “ º ” (letra “o” minúscula sobrescrita) ou como “ ° ” (símbolo de grau – código 176). A palavra abreviada “números” deve ser codificada como “nºs” utilizando a letra “s” sem o sobrescrito ou sublinhado.

De acordo com o texto da Lei Complementar nº 95, os incisos são codificados em número romanos e separados utilizando o hífen “-” (código 45) e não por símbolos como *ene* “ – ” (código 150) ou *eme* “ — ” (código 151).

A codificação errada de caracteres pode gerar vários problemas, como, por exemplo, erro na pronúncia por parte dos softwares de síntese de voz utilizados por cegos ou pessoas com baixa visão.

5. Visão Geral do esquema LexML

A Parte Inicial apresenta construtores (grupos, tipos, etc.) que são utilizados pela Parte Principal. Como a versão foi derivada a partir do *XML Schema* AKOMA NTOSO, foram preservados a forma de organização e alguns comentários originais. Alguns construtores da parte inicial estão definidos com nome em inglês, no entanto estes nomes são utilizados apenas internamente no esquema e não são visíveis nas instâncias de documentos LexML.

5.1. LexML - Parte Inicial

A parte inicial do esquema relaciona alguns grupos de elementos (`<xsd:groups>`) e atributos (`<xsd:attributeGroup>`) utilizados na definição dos modelos de conteúdos e tipos em todo o documento. Na seqüência, são definidos tipos simples (`<xsd:simpleType>`), basicamente para enumeração de valores *string*, e tipos complexos (`<xsd:complexType>`). Esses tipos complexos correspondem a *Design Patterns* de modelo de conteúdo referenciados na definição dos elementos. Para detalhes sobre estes e outros *Design Patterns*, consultar (Vitali et. al, 2005) e (Vitali, 2006).

Hierarchy –elementos de estrutura hierárquica (ex: `<Livro>`, `<Parte>`, etc.).

Blocks – seqüência de elementos bloco (ex: `<p>`).

Inline – trata elementos conteúdo misto (ex: conteúdo de `<p>`).

Marker – elemento de conteúdo vazio (ex: `<NotaReferenciada/>`).

O *Design Pattern Container* não possui um tipo complexo associado a exemplo dos outros. Cada elemento desta categoria organiza os elementos na ordem desejada utilizando os elementos construtores da linguagem *XML Schema*.

Para mais informações técnica sobre os cinco *Design Patterns* citados, consultar a Seção 4 do documento "*Akoma Ntoso* 1.0 *Release Notes*" (Vitali, 2007).

5.2. LexML - Parte Principal

Todos os documentos LexML compartilham o mesmo elemento raiz `<LexML>`, cujo conteúdo inicial é obrigatoriamente o elemento `<Metadado>` seguido do elemento que identifica o tipo de documento. A utilização de um único elemento raiz segue o *Design Pattern "Universal Root"*.

O LexML prevê inicialmente os seguintes tipos de documentos:

`Norma` – Constituição, Emendas, Leis, Decretos etc.

`ProjetoNorma` – Proposições legislativas.

`Jurisprudencia` – Súmulas e Acórdãos.

`DocumentoGenerico` – Para outros tipos de documentos.

`Anexo` – Anexos dos documentos acima.

A Norma e o Projeto de Norma utilizam o tipo `HierarchicalStructure`; os Anexos (Norma e Projeto) podem ser definidos utilizando tanto uma estrutura hierárquica quanto uma estrutura genérica, dependendo do caso específico. A Jurisprudência utiliza a estrutura genérica para cada um dos seus componentes textuais (Relatório, Voto, etc).

Na seção seguinte do esquema, são definidos *containers* referenciados na definição dos tipos de documento. Por exemplo, é possível encontrar nesta seção os elementos `ParteInicial`, `Articulacao` e `ParteFinal`, referenciados na definição do elemento `Norma`.

Os elementos que definem a hierarquia de um documento articulado, de acordo com a Lei Complementar nº 95, são definidos na seção seguinte. Inicialmente são apresentados os agrupadores de artigo (Parte, Livro, Titulo, Capitulo, Secao e Subsecao), em seguida a unidade básica de articulação (Artigo), e, por fim, os dispositivos de artigo (Caput, Inciso, Alinea, Item, Paragrafo).

Na seqüência, são apresentados os elementos genéricos correspondente a cada um dos cinco *patterns* de modelo de conteúdo. Agrupamento (Container), AgrupamentoHierarquico (Hierarchy), Bloco (Block), EmLinha (Inline), Marca (Marker). Ao se utilizar estes elementos deve-se especificar um atributo nome significativo.

Para mais informações técnica sobre os elementos genéricos, consultar a Seção 5 do documento "*Akoma Ntoso* 1.0 *Release Notes*" (VITALI, 2007).

Em seguida, o esquema apresenta os elementos "importados" do vocabulário HTML que tratam de texto, listas, tabelas e outros elementos de apresentação como negrito, itálico e sublinhado. Trata-se de uma simplificação do vocabulário HTML.

De forma diferente do AKOMA NTOSO, para marcar algum termo que está sendo definido nos textos, utilizamos a marcação <dfn> e não <def>, por ser a primeira parte do vocabulário HTML.

Por fim, são apresentados os elementos de metadados que tratam de vários aspectos tais como a indexação, o ciclo de vida e informações sobre a publicação.

5.3. Identificadores de Agrupamento de Artigo

A definição de identificadores para os dispositivos tem por objetivo identificar partes de um documento. No esquema LexML, a exemplo do que ocorre no AKOMA NTOSO, os identificadores são permitidos em todos os elementos, e, em alguns, devido a sua importância, são atributos obrigatórios. Eles podem ser utilizados, por exemplo, nas remissões textuais de um dispositivo ou na indicação de qual dispositivo está sendo alterado.

Os identificadores de Agrupadores de Artigos são criados a partir da concatenação do identificador de cada nível hierárquico desde o nível mais alto até o nível que se está identificando. Por exemplo, `id="tit1_cap2-A_sec3"` é o identificador da Seção III do Capítulo II-A do Título I. Note que os algarismos romanos são convertidos em números arábicos e os sufixos "-A" resultante da inclusão de dispositivos à norma permanecem com o(s) hífen(s) e a(s) letra(s). No caso do elemento `<Parte>`, normalmente identificado por um nome (Parte Geral, Parte Especial), utiliza-se um seqüencial a partir do número 1.

A Tabela 2 apresenta os identificadores de agrupamento de artigo previstos na Lei Complementar nº 95, bem como o elemento genérico `<AgrupamentoHierarquico>` que poderá ser utilizado nos casos não previstos. Em algumas situações será necessário utilizar o esquema "LexML flexível" que permite uma maior mais combinações dos agrupadores de artigos.

Tabela 2. Identificadores agrupadores de Artigo.

Elemento	id	Exemplo	id do exemplo
<Parte>	prtN	Parte Geral do Código Civil	prt1
<Livro>	livN	Livro I	liv1
<Titulo>	titN	Título II	tit2
<Capitulo>	capN	Título IX, Capítulo IV-A	tit9_cap4-A
<Secao>	secN	Seção III do Capítulo I	cap1_sec3
<Subsecao>	subN	Parte Geral do Código Civil, Livro I, Título VI, Capítulo I, Seção II, Subseção I	prt2_liv1_tit6_cap1_sec2_sub1
<AgrupamentoHierarquico>	aghN	Subtítulo I do Título II do Livro II da Parte Especial do Código Civil	prt2_liv2_tit2_agh1

5.4. Identificadores de Artigos e Dispositivos de Artigo

A lógica de formação dos identificadores de artigos é similar à dos agrupadores, atentando-se ao fato da presença do elemento <Caput> que não possui o seqüencial numérico por ele ser único e obrigatório.

A Tabela 3 apresenta os identificadores de artigo e seus dispositivos previstos na Lei Complementar nº 95, bem como o elemento genérico <DispositivoGenerico> que poderá ser utilizado nos casos de dispositivos não previstos ou em desacordo com os da lei complementar. Em algumas situações será necessário utilizar o esquema “LexML flexível" que permite mais combinações entre os dispositivos de artigos. A tabela mostra também exemplos em que letras das alíneas são convertidas em números arábicos.

Tabela 3. Identificadores de Artigo e seus Dispositivos.

Elemento	Id	Exemplo(s)	id do exemplo
<Artigo>	artN	Art. 5º	art5
<Caput>	cpt	caput do art. 5º	art5_cpt
<Paragrafo>	parN	Art. 8-A, parágrafo único	art8-A_par1
<Inciso>	incN	inciso III do art 8º inciso II do § 2º do art 20	art8_cpt_inc3 art20_par2_inc2
<Alinea>	aliN	Alínea “a” do inciso IX do parágrafo único do art. 3º-A-A.	art3-A-A_par1_inc9_ali1
<Item>	iteN	item 1, da alínea “b”, do inciso I, do art. 39	art39_cpt_inc1_ali2_ite1
<DispositivoGenerico>	dpgN	alínea “b” do § 3º do art. 5º	art5_par3_dpg2

Na seção de definição dos tipos simples do esquema LexML, foram definidos *patterns* com expressões regulares para validação dos identificadores.

5.5. Outros Identificadores

Os identificadores servem de "gancho" para a localização de determinado fragmento do documento. No caso de normas jurídicas, foram definidos alguns identificadores para rápida localização de outros segmentos de texto, tais como: `epigrafe`, `ementa`, `preambulo`, `localdata` e `fecho`.

Os identificadores de dispositivos alvo de uma alteração (normalmente o novo texto destes dispositivos aparecem entre aspas) são precedidos pelo prefixo `CONT_aspN_`, onde `CONT (contexto)` representa o id do dispositivo da alteração na norma publicada e `N` representa o seqüencial das aspas referente à alteração. Por exemplo, o identificador `art14_cpt_asp1_art4_cpt_inc3` identifica uma alteração do Inciso III do art. 4º da norma identificada na tag <Alteracao>. Este dispositivo está localizado na primeira seqüência de aspas do caput do art 14 da norma alteradora.

6. Estrutura de Acórdão em LexML

O inteiro teor de um Acórdão é formado por seis elementos conforme a seguir:

- <CabecalhoAcordao> - contém dados estruturados comuns como epígrafe, data de julgamento, órgão julgador, partes além do texto da ementa. A informação sobre publicação/republicação será codificada no elemento <Metadado>.
- <AcordaoTexto> - texto livre com o acórdão.
- <RelatorioTexto> - texto livre com o relatório.
- <VotoTexto> - texto livre com o voto.
- <DebateTexto> - texto livre opcional com debates.
- <ExtratoAtaTexto> - texto livre com o Extrato da Ata.

A estrutura para texto livre do LexML permite a codificação de texto com sofisticadas características, tais como definição de links persistentes para normas (de qualquer esfera) e julgados.

As notas de rodapé são tratadas em cada uma dos componentes de texto do acórdão utilizando o elemento <`div`> e links bidirecionais. Dessa forma é possível isolar este elemento ao mesmo tempo em que permite a navegação entre a nota e o texto principal.

O elemento "*class*" pode ser codificado em cada parágrafo para tratar as questões de apresentação em uma folha de estilo CSS.

7. Tratamento do Texto

Com o intuito de facilitar o desenvolvimento de ferramentas para redação de documentos no formato LexML, utilizou-se dois tipos de construtores para conter o texto propriamente dito:

- `<Texto>` - elemento tipo *container*, que admite vários blocos (`<p>, <table>, <ol>, <ul> etc`)
- `<TextoSimples>` - elemento tipo *bloco*, permitindo conteúdo equivalente a de um parágrafo `<p>`.

O texto propriamente dito é marcado utilizando um subconjunto de elementos HTML. Como falado anteriormente, os elementos importados do HTML fazem parte do mesmo *namespace* do LexML, para facilitar a gerência de *namespaces*.

Os elementos importados desempenham o mesmo papel daquele definido para o HTML. Acrescenta-se a esta regra apenas uma exceção: o elemento `<div>`, que no HTML desempenha o papel de bloco genérico, no LexML, desempenha o papel de *container* genérico, pois o elemento `<p>` já desempenha o papel de bloco genérico.

Os elementos `<div>`, `<p>` e `<span>` são considerados elementos genéricos de *container*, *bloco* e *inline* respectivamente, sendo equivalentes a `<Agrupador>`, `<Bloco>` e `<EmLinha>`.

Dentro do elemento <Texto> ou `<TextoSimples>` é possível utilizar o elemento *inline* <Remissao> para criar referências *href* para outros documentos do LexML.

As referências poderão ser absolutas ou relativas. Este último caso ocorre quando o elemento `<Remissao>` está codificado internamento ao elemento `<Alteracao>`: no elemento remissão será codificado a referência relativa para o *id* do dispositivo apontado (`href="#art1_par1"`), enquanto que no elemento `<Alteracao>` haverá a referência absoluta para a norma (`base="urn:lex:br;sp:lei:2004-02-11;123"`).

Para otimizar a marcação de referências, é possível utilizar o elemento <RemissaoMultipla> que conterá mais de um elemento <Remissao>. O primeiro utiliza o atributo *base*, e o segundo utiliza o atributo *href* com referências relativas.

Todos os elementos do LexML (inclusive os "importados") podem ser marcados com atributos *class* e *style*, permitindo a associação de estilos CSS com definições precisas de apresentação.

8. Metadados

O elemento Metadado contém informações sobre o documento para atender diversas necessidades tais como recuperação da informação, preservação e gerência do ciclo de vida do documento.

Como regra, todo conteúdo feito por um editor (e não pelo autor), deve fazer parte desta seção. Vice versa, todo conteúdo do autor, não deve ser codificado na seção de metadados.

A Seção <Metadados> está estruturada em oito subseções, conforme a seguir:

- Identificação – contém elementos de metadados que identificam unicamente o arquivo corrente.
- Contexto – identifica a posição do arquivo corrente na hierarquia FRBR. A posição é indicada pelo atributo ‘ id=“self” ’ do elemento FRBRItem.
- CicloDeVida – registra os eventos relacionados ao arquivo corrente. No caso de textos multivigente, deve-se registrar o histórico de eventos do período relativo à vigência do documento.
- EventosGerados – registra os eventos de alteração que afetam outros documentos. Por exemplo, a alteração e revogação de dispositivos de outras normas.
- Notas – registra notas do editor ou do responsável pelo markup. As notas podem ser visíveis, nos casos em que registre informações sobre a versão corrente, ou invisíveis, nos casos em que faz-se um registro técnico sobre o markup.
- Recursos – lista alguns recursos auxiliares que podem referenciados em outras pontos da seção de metadados.
- MetadadosProprietario – ponto de extensão para registro de metadados não considerados no atual esquema.

Para o detalhamento dos campos da Seção Identificação, deve-se consultar o documento LexML_03_URN e, para o entendimento da hierarquia FRBR deve-se consultar o documento LexML_01_ModReferencia.

9. Referências

9.1. Bibliográficas

LEAL, A. *Technica Constitucional Brazileira*. Rio de Janeiro: Typ. Do Jornal do Commercio, de Rodrigues & C. 1914. 81 p.

VITALI, F; Di IORIO, A.; GUBELLINI, D.; Design Patterns for Descriptives Document Substructures. *In Extreme Markup Language 2005*. Montreal, Québec. 2005. Acessado em 03 de Janeiro de 2007. Disponível em: [www.mulberrytech.com/Extreme/Proceedings/html/2005/Vitali01/EML2005Vitali01.html]

VITALI, F. *Akoma Ntoso 1.0 Release Notes*. (versão 23/11/2006). 2006. Acessado em 3 de fevereiro de 2007. Disponível em: [http://www.akomantoso.org/ AKOMA_NTOSO/ downloads-1/Release_Notes_20112006.pdf]

9.2. Sites

http://www.nir.it/

Projeto Norme in Rete (Itália)

http://www.akomantoso.org/

Projeto Akoma Ntoso (Nações Unidas/África)

http://www.xmlpatterns.com/

Design Patterns para XML

http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=22109&ICS1=1&ICS2=140&ICS3=20

Padrão ISO 639-2 - Codes for the representation of names of languages -- Part 1: Alpha-2 code

U