RELAX NG to DTD and XSD using the Open Toolkit

Using RELAX NG for DITA shells and modules

Eliot Kimber
Contrext, LLC

DITA OT Day Munich 2014

About the Author

- Responsible for DITA 1.3 grammar files
- Independent consultant focusing on DITA analysis, design, and implementation
- Doing SGML and XML for cough 30 years cough
- Founding member of the DITA Technical Committee
- Founding member of the XML Working Group
- Co-editor of HyTime standard (ISO/IEC 10744)
- Primary developer and founder of the DITA for Publishers project
- Author of DITA for Practitioners, Vol 1 (XML Press)

Agenda

- A brief demonstration
- What is RELAX NG?
- RELAX NG and DITA
- DITA document types and modules
- Anatomy of a RELAX NG shell
- See how easy: generating DTD and XSD shells

Executive Summary

- RELAX NG makes XML vocabulary definition much, much easier
- RELAX NG is a good match to DITA requirements
- Can generate conforming DITA DTD and XSD files from RELAX NG
- Makes creating DITA document type shells about as easy as it can be
- No more need to fight with DTD or XSD syntax

Demonstration

- Create a new document type shell
- Create a new domain specialization
- Create a new topic speicalization
- Generate DTD and XSD versions of all of these grammars

WHAT IS RELAX NG?

RELAX NG is

- A grammar (or schema) language for XML documents
- An OASIS and ISO/IEC standard (ISO/IEC 19757-2)
- Most like XML DTDs but easier to use
- Designed to be as simple as possible
- Used for other XML standards, most prominently DocBook

RELAX NG Basics

- RELAX NG has both an XML and character-based syntax:

```
<grammar>
  <start>
    <ref name="concept.element"/>
  </start>
  …
</grammar>
```

```
grammar {
  start concept.element
  …
}
```

- The two are functionally equivalent

RELAX NG Basics (cont.)

- A RELAX NG grammar defines a set of "patterns" that documents must match
- Patterns can have names (analagous to DTD parameter entities)
- Pattern combination mechanisms are more flexible than DTD parameter entities or XSD groups
- Grammars can be modularized into multiple files

RELAX NG AND DITA

Good Match to DITA Requirements

- RELAX NG's pattern-based approach works well with DITA's modular vocabulary
- Patterns can allow extension by other patterns
- Patterns can unilaterally extend other patterns: the extending pattern names the pattern it extends
- Makes combining vocabularies as easy as it could be

No Added Overhead

- Unlike DTDs, RELAX NG has much simpler and easier-to-use syntax
- Avoids the conceptual and syntactic complexity of XSD
- Close semantic match to existing DTD-defined rules

Does require DTD Compatibility Feature

- Base RELAX NG standard does not provide for defining attribute defaults
- Companion standard *RELAX NG DTD Compatibility* provides attribute defaults
- DITA depends on default attribute values
- George Bina (oXygenXML) has implemented DTD compatibility in Java for use with Xerces XML parser (open source)

Result: Everything is Awesome

- RELAX NG makes DITA document type shells as easy to create as it can be
- Can generate DITA-conforming DTDs and XSDs from the RELAX NG grammars
- Makes it possible, if not in fact easy, for anybody to configure their own DITA document types

DITA DOCUMENT TYPES AND SHELLS

DITA Document Types

- A DITA document type is a unique set of vocabulary and constraint modules
- Modules are *invariant*
  - Every copy of a given module should be identical
  - You *never* modify modules directly
- You implement DITA document types by *integrating* vocabulary and constraint modules using a *document type shell*

Document Type Shells

- *Integrate* modules to form a working document schema
- Refer to the vocabulary and constraint modules of the DITA document type
- Do any *configuration* required
- Two document type shells that integrate the same same set of modules implement the same DITA document type

ANATOMY OF A RELAX NG SHELL

Shell for Base Topic Document Type (basetopic.rng)

```
<grammar xmlns=http://relaxng.org/ns/structure/1.0
 xmlns:dita="http://dita.oasis-open.org/architecture/2005/"
 xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0">
  …
 <div>
    <a:documentation>ROOT ELEMENT DECLARATION</a:documentation>
    <start combine="choice">
      <ref name="topic.element"/>
    </start>
  </div>
  <div>
    <a:documentation>DOMAINS ATTRIBUTE</a:documentation>
    <define name="domains-att">
      <attribute name="domains"
        a:defaultValue="(topic topic) (topic hazard-d) (topic hi-d) (topic indexing-d)
                        (topic ut-d) (topic hazard-d)"/>
    </define>
  </div>
  <div>
    <a:documentation>MODULE INCLUSIONS</a:documentation>
    <include href="topicMod.rng">
      <define name="topic-info-types">
        <ref name="topic.element"/>
      </define>
    </include>
    <include href="hazardstatementDomainMod.rng"/>
    <include href="highlightDomainMod.rng"/>
    <include href="indexingDomainMod.rng"/>
    <include href="utilitiesDomainMod.rng"/>
  </div>
  …
</grammar>
```


Integration: Just Include the Module

- All you do is include the module you want to integrate:

```
<include href="topicMod.rng">
  <define name="topic-info-types">
    <ref name="topic.element"/>
  </define>
</include>
<include href="hazardstatementDomainMod.rng"/>
<include href="highlightDomainMod.rng"/>
<include href="indexingDomainMod.rng"/>
<include href="utilitiesDomainMod.rng"/>
```


Modules are Self Integrating

- Unilaterally extend the base patterns (from highlightDomainMod.rng):

```
<define name="hi-d-ph">
  <choice>
    <ref name="b.element"/>
    <ref name="i.element"/>
    <ref name="sup.element"/>
    <ref name="sub.element"/>
    <ref name="tt.element"/>
    <ref name="u.element"/>
  </choice>
</define>

<define name="ph" combine="choice">
  <ref name="hi-d-ph"/>
</define>
```

- Only need to do something special when you need to impose constraints

Compare With DTD

```
<!-- ============================================================= -->
<!--            DOMAIN ENTITY DECLARATIONS                         -->
<!-- ============================================================= -->

<!ENTITY % hazard-d-dec
  SYSTEM "hazardstatementDomain.ent"
>%hazard-d-dec;

<!ENTITY % hi-d-dec
  SYSTEM "highlightDomain.ent"
>%hi-d-dec;

<!ENTITY % indexing-d-dec
  SYSTEM "indexingDomain.ent"
>%indexing-d-dec;

<!ENTITY % ut-d-dec
  SYSTEM "utilitiesDomain.ent"
>%ut-d-dec;

<!-- ============================================================= -->
<!--            DOMAIN ATTRIBUTES DECLARATIONS                     -->
<!-- ============================================================= -->

<!-- ============================================================= -->
<!--                  DOMAIN EXTENSIONS                            -->
<!-- ============================================================= -->
<!--                   One for each extended base element, with
                      the name of the domain(s) in which the
                      extension was declared                       -->

<!ENTITY % note        "note |
                        %hazard-d-note;
                        ">
<!ENTITY % ph          "ph |
                        %hi-d-ph;
                        ">
<!ENTITY % index-base   "index-base |
                        %indexing-d-index-base;
                        ">
<!ENTITY % fig         "fig |
                        %ut-d-fig;
                        ">

<!-- ============================================================= -->
<!--                  DOMAIN ATTRIBUTE EXTENSIONS                  -->
<!-- ============================================================= -->

<!ENTITY % props-attribute-extensions
  ""
>
<!ENTITY % base-attribute-extensions
  ""
>
```

```
<!-- ============================================================= -->
<!--                  TOPIC NESTING OVERRIDE                       -->
<!-- ============================================================= -->

<!ENTITY % topic-info-types
          "topic"
>

<!-- ============================================================= -->
<!--                  DOMAINS ATTRIBUTE OVERRIDE                   -->
<!-- ============================================================= -->

<!ENTITY included-domains
                      "&hazard-d-att;
                       &hi-d-att;
                       &indexing-d-att;
                       &ut-d-att;
  "
>

<!-- ============================================================= -->
<!--                  CONTENT CONSTRAINT INTEGRATION               -->
<!-- ============================================================= -->

<!-- ============================================================= -->
<!--                  TOPIC ELEMENT INTEGRATION                    -->
<!-- ============================================================= -->

<!ENTITY % topic-type
  SYSTEM "topic.mod"
>%topic-type;

<!-- ============================================================= -->
<!--                  DOMAIN ELEMENT INTEGRATION                   -->
<!-- ============================================================= -->

<!ENTITY % hazard-d-def
  SYSTEM "hazardstatementDomain.mod"
>%hazard-d-def;

<!ENTITY % hi-d-def
  SYSTEM "highlightDomain.mod"
>%hi-d-def;

<!ENTITY % indexing-d-def
  SYSTEM "indexingDomain.mod"
>%indexing-d-def;

<!ENTITY % ut-d-def
  SYSTEM "utilitiesDomain.mod"
>%ut-d-def;
```


SEE HOW EASY: GENERATING DTD AND XSD SHELLS

RNG-to-DITA DTD and XSD Converters

- Generate conforming DTD and XSD shells and modules from DITA RELAX NG modules
- Require DITA-specific metadata in the shells and modules
- Maintained by the DITA Community under direction of the DITA TC
  - Eliot writes and maintains the code, the TC determines what the code needs to do to be correct and complete.

DITA OT RNG Conversion Plugin

- TC currently provides:
  - OT plugin for DITA 1.3 grammars: org.oasis-open.dita.dita13.doctypes
  - OT plugin for RNG conversion: org.oasis-open.dita.rng.converter
- Defines transformation types:
  - rng2dtd
  - Rng2all
- Requires Saxon 9.6+

File Organization Requirements

- RNG files must be organized in the same way as the TC-provided files:
  - Top-level directory to hold all grammars types:
    - doctypes/
  - Directory named “rng/” for RNG grammars:
    - doctypes/rng
  - Directory for each group of grammars (e.g., one per shell):
    - doctypes/rng/indirector
  - Directory named “rng” under that:
    - doctypes/rng/indirectory/rng

Running The DTD Generator

- Use transtype “rng2dtd”
- Apply transform to file in parent directory containing all RNG modules:
  - E.g., catalog.xml in your doctypes/rng/ directory
- By default, only generates shells

Plugin Parameters

- generateModules=true|false
  - Default is “false” (only generate shells)
- usePublicIDsInShell=true|false
  - Default is true
- generateCatalogs=true|false
  - Default is true

Resources

- RELAX NG specifications: http://relaxng.org
- DITA 1.3 spec: http://tools.oasis-open.org/version-control/svn/dita/spec/
- DITA RELAX NG conversion tools: http://tools.oasis-open.org/version-control/svn/dita/doctypes/tools/relaxng/
- Me: ekimber@contrext.com, http://contrext.com