An OASIS DITA Adoption Technical Committee Publication

DITA 1.2 Feature Article:
Convenience elements for map references

Bruce Nevin
On behalf of the OASIS DITA Adoption Technical Committee

Date: 30 June 2011

OASIS (Organization for the Advancement of Structured Information Standards) is a not-for-profit, international consortium that drives the development, convergence, and adoption of e-business standards. Members themselves set the OASIS technical agenda, using a lightweight, open process expressly designed to promote industry consensus and unite disparate efforts. The consortium produces open standards for Web services, security, e-business, and standardization efforts in the public sector and for application-specific markets. OASIS was founded in 1993. More information can be found on the OASIS website at http://www.oasis-open.org.

The OASIS DITA Adoption Technical Committee members collaborate to provide expertise and resources to educate the marketplace on the value of the DITA OASIS standard. By raising awareness of the benefits offered by DITA, the DITA Adoption Technical Committee expects the demand for, and availability of, DITA conforming products and services to increase, resulting in a greater choice of tools and platforms and an expanded DITA community of users, suppliers, and consultants.

DISCLAIMER: All examples presented in this article were produced using one or more tools chosen at the author's discretion and in no way reflect endorsement of the tools by the OASIS DITA Adoption Technical Committee.

This white paper was produced and approved by the OASIS DITA Adoption Technical Committee as a Committee Draft. It has not been reviewed and/or approved by the OASIS membership at-large.

Document History

Revision	Date	Author	Summary

Table of Contents

Convenience elements for map references

> *A number of "convenience elements" have been specialized in DITA 1.2 to simplify the work of creating and maintaining maps and to make it easier for a map author to read and understand them.*

The <topicref> element, as its name implies, was originally designed for building a complex document by referring to topics from a map. However, <topicref> can also be used for other purposes if the appropriate attribute values are set. For example, a <topicref> element can point to another map rather than to a topic if the @format attribute is set to "ditamap".

As a convenience, a number of elements have been specialized from <topicref> that differ from it only in having default values set for those attributes — and of course the name of the specialized element is different. This simplifies the map author's work, and makes it obvious at a glance which references in a map point to topics and which ones have other functions, such as pointing to submaps.

These are the six convenience elements discussed in this paper:

- **<mapref>** includes an entire map by reference.
- **<topicset>** defines a group of <topicref> elements as a referenceable branch of a map.
- **<topicsetref>** includes a submap by reference to a branch of a map defined by the <topicset> element.
- **<topicgroup>** specifies attributes and linking characteristics to be shared by a group of <topicref> elements in a map.
- **<anchor>** indicates where content is to be pushed at run time.
- **<anchorref>** pushes content to an <anchor> element at run time.

Five of these are new with DITA 1.2. Even though the <topicgroup> element is not new, it is useful to review all six of them together.

There are several other convenience elements that are not discussed here. The <keydef> element is also specialized from <topicref>, and it is new with DITA 1.2, but it has a very different purpose (defining keys for key references) which is beyond the scope of this paper. The <glossref> element is a Technical Content specialization for the construction of a glossary; it is adequately described in the Language Reference. Finally, <topichead> inserts a title into a map without any immediately following content; it is not new in DITA 1.2.

We'll consider our six convenience elements under these headings:

- *Submaps: <mapref>, <topicsetref> & <topicset>*
- *Group management of linking and inheritance: <topicgroup>*
- *Run-time integration: <anchor> & <anchorref> or @anchorref*

Use cases and examples illustrate some of the purposes to which the map convenience elements can be put, and the tools support that will be needed to make full use of them.

At the end of the paper is a brief discussion of how these elements are processed in *relationship tables*.

Submaps: <mapref>, <topicsetref> & <topicset>

Use the <mapref> element to merge the entire hierarchy of another map into the current map, including its relationship tables.

Use the <topicssetref> element to incorporate just part of a map into the current map. The <topicssetref> element points to a <topicset> element, which delimits a contiguous subset of the <topicref> elements in a map. The content within a <topicset> element is always rendered at the location of that element. When a map that contains <topicset> is processed, the <topicset> element is effectively invisible, as though it were not there. It functions as a sub-branch of the map only when a <topicssetref> element refers to it. The referenced <topicset> element can be in the same map or in another map.

Note:

When a <topicsetref> element is placed in a relationship table, just one link is generated, pointing to the root element of the document that is delimited by the referenced <topicset> element. Consequently, a <topicset> element that is referenced by a <topicsetref> element within a <relcell> element should have just one child element with any additional topic references hierarchically subordinated under it. Depending on your reuse discipline, this may be a best practice for you to generalize to all instances of the <topicset> element. If the <topicset> element contains a sequence of topic references that are peers (not subordinated under one topic reference in a hierarchy), you probably expect to see a link to each of those topics, but only one link to the first topic of the sequence will be rendered in the relationship table. The <topicgroup> element delimits a collection that is not organized in a rooted hierarchy. When an <topicgroup> is placed in a relationship table, links to each contained topic are rendered severally.

Example of <mapref>

A team documenting a development tool needs to include the documentation for the class libraries that are bundled with the development tool. The team uses a <mapref> element to import the maps for the class libraries into the map for the development tool. The main map looks like this:

```
<map>
   <title>Programming With BetterWidgetMaker</title>
   <topicref href="bwm-overview.dita>
      <topicref href="bwm-data-structures.dita"/>
      <topicref href="bwm-io.dita/>
   </topicref>
   <mapref href"bwm-libraries.ditamap"/>
 <topicref href="debug.dita/>
</map>
```

The referenced bwm-libraries.ditamap is as follows:

```
<map>
   <topicref href="libraries.dita>
      <topicref href="statlib.dita>
      <topicref href="codegenlib.dita"/>
      <topicref href="dll.dita/>
   </topicref>
</map>
```

A reader of "Programming with BetterWidgetMaker" sees the published content on libraries integrated with the preceding chapters as though the main map were organized this way:

```
<map>
   <title>Programming With BetterWidgetMaker</title>
   <topicref href="bwm-overview.dita>
      <topicref href="bwm-data-structures.dita"/>
      <topicref href="bwm-io.dita/>
   </topicref>
 <!-- The included submap bwm-libraries.ditamap begins here -->
   <topicref href="libraries.dita>
      <topicref href="statlib.dita>
      <topicref href="codegenlib.dita"/>
      <topicref href="dll.dita/>
   </topicref>
 <!-- The included submap ends here -->
<topicref href="debug.dita/>
</map>
```

Example of <topicset> and <topicsetref> for curriculum development

A curriculum developer creates a collection of course material topics about SQL. The lesson on SQL basics is presented in several topics about selecting rows from a database in a map named sqlbasics.ditamap:

```
<map>
<!-- sqlbasics.ditamap -->
 ...
 <topicref href="sqlSelection.dita"/>
 <topicref href="sqlJoin.dita"/>
 <topicref href="sqlFilter.dita"/>
 ...
</map>
```

To reuse that lesson in a course about mastering JDBC, the curriculum developer modifies sqlbasics.ditamap by putting those lesson topics in a <topicset>:

```
<map>
<!-- sqlbasics.ditamap -->
 ...
 <topicset id="sqlbasics" href="sqlOverview.dita">
   <topicref href="sqlSelection.dita"/>
   <topicref href="sqlJoin.dita"/>
   <topicref href="sqlFilter.dita"/>
   ...
 </topicset>
 ...
</map>
```

This defines a branch of sqlbasics.ditamap, a submap which can then be referenced with a <topicsetref> in the Mastering JDBC course:

```
<map>
<!-- masteringjbc.ditamap -->
 ...
 <topicsetref href="sqlOverview.dita#sqlbasics"/>
 <topicref href="jdbcPrepare.dita"/>
 ...
</map>
```

The result is as though the topics in the <topicset> were copied to the location of the <topicsetref>:

```
<map>
<!-- masteringjbc.ditamap -->
 ...
 <topicref href="sqlSelection.dita"/>
 <topicref href="sqlJoin.dita"/>
 <topicref href="sqlFilter.dita"/>
 ...
 <topicref href="jdbcPrepare.dita"/>
 ...
</map>
```

A learner masters the course about SQL basics, including the lesson on selecting rows. The LMS (Learning Management System) records the lessons taken by the learner. Later, the learner starts the course on JDBC. The LMS recognizes that the learner has already read the lesson on selecting rows and alerts the learner to that fact so the learner can confidently skip the familiar portion of the course.

To facilitate such reuse, the curriculum developer organizes all the lesson material in the SQL course into topics grouped with <topicset> elements. The map for the entire course, sqlcourse.ditamap, looks like this:

```
<map>
<!-- sqlcourse.ditamap -->
 <topicref href="sqlIntro"/>
 <topicsetref href="sqlsyntax"/>
 <topicsetref href="sqlbasics"/>
 <topicsetref href="sqlstatements"/>
 <topicsetref href="sqlclauses"/>
 <topicsetref href="sqloperators"/>
 ...
</map>
```

Example of <topicset> and <topicsetref> for intelligent search

Consider an investment company that provides trends and profiles to its brokers. Each trend or profile is authored as a sequence of topics contained in a <topicset> element. When a broker searches for information, an intelligent search engine recognizes when the distribution of hits indicates a good match on a <topicset> as a whole. The search engine returns the equivalent of a small booklet generated from that <topicset>, distinguishing it from individual topics located individually or scattered through other <topicset> collections.

Group management of linking and inheritance: <topicgroup>

When <topicref> elements (or specializations of <topicref>) are nested within a <topicref> element, they inherit attribute values from the containing <topicref>. They are also subordinate in the structural hierarchy, in the manner of

a section and subsections. You may want the efficiency of managing attribute values for a set of topics in one place without imposing a structural hierarchy.

Use the <topicgroup> element to specify attribute values for all the contained <topicref> elements without creating a hierarchy. For details about how child elements of <topicgroup> inherit attribute values, see 2.1.2.3.4 "Cascading of attributes and metadata in a DITA map" and 2.1.2.3.5 "Map-to-map cascading behaviors" in the DITA 1.2 specification.

When a <topicgroup> element is placed in a relationship table, links are made to all topics referenced by the contained <topicref> elements.

Note:

This element is not new with DITA 1.2.

Example using <topicgroup>

In this example, a course developer has decided that the introduction, the basic SQL operations, and SQL statements and clauses, are material suitable for a novice level audience. Putting them in a <topicgroup> element makes them available as a group for inclusion in any document for novice users, and when the current map is published for a non-novice audience this entire group can be excluded.

```
<map>
<!-- sqlcourse.ditamap -->
 <topicgroup audience="novice" linking="none">
  <topicref href="sqlIntro"/>
  <topicsetref href="sqlsyntax"/>
  <topicsetref href="sqlbasics"/>
  <topicsetref href="sqlstatements"/>
  <topicsetref href="sqlclauses"/>
 </topicgroup>
 <topicsetref href="sqloperators"/>
 ...
</map>
```

Each <topicref> and <topicsetref> element within the <topicgroup> inherits the values of the audience and linking attributes from it. In this way the course developer only has to set these common attributes in this one place in the map to affect all of the referenced topics within the scope of the <topicgroup> element. The other topics referenced in sqlcourse.ditamap are unaffected. Adjustments as to which topics are included and which are excluded can be made at one place in the map rather than in each <topicref> element or in each topic, and the same topics can be assigned different metadata in another map.

Run-time integration: <anchor> and <anchorref> or @anchorref

Use the <anchor> element to specify a location within a map at which content may be integrated at run time, as for Eclipse Help. This can be done in two ways.

- The @anchorref attribute on a map pushes the entire content of the map to the specified <anchor> location.
- The <anchorref> element within a map delimits a contiguous subset of <topicref> elements as a branch of the map to be pushed to the location of the <anchor> element.

The <anchorref> element may be thought of as the push equivalent of the <topicset> element. Like the <topicset> element, it defines a branch of the containing map. As with the <topicset> and <topicsetref> elements, the content within <anchorref> is still rendered where it is located; in addition, with appropriate processing, the content within the <anchorref> element is also rendered at the location of the <anchor> element whose @id attribute value it specifies. Consequently, an <anchorref> element should have just one child element with any additional topic references hierarchically subordinated under it.

Note:

This behavior requires processing support that is not always present. For build-time integration of content, use the @conref or @conkeyref attribute on an element within the map.

Note:

The <anchor> element should not be used in a relationship table. Like the <navref> element, it cannot be resolved until content is being displayed.

Note:

The similarity of the names <anchorref> and <topicsetref> may be confusing. The <anchorref> element defines a map branch, but the <topicsetref> element refers to a map branch that is defined elsewhere. With <topicsetref> as the referring element, content is pulled from the referenced location; with <anchorref> as the referring element, content is pushed to the referenced location. This is exactly parallel to conref push, where the referring element contains the content to be pushed to the referenced element.

Example of <anchor>, @anchorref, <anchorref>

A content architect for a car rental company has organized a map called carprep.ditamap to define the end-to-end process of preparing a car for rental. Some instructions on preparing a car for the next rental are the same for all makes, and some are specific to a brand and model. The map has <anchor> elements at appropriate extension points for model-specific tasks as part of the process.

Another writer has written a series of maps defining the tasks specific to maintaining each model of car in the rental fleet. These model-specific maps include scheduled maintenance and other tasks that are not necessary for a quick turnaround between rentals. The map for the Astro, called astro.ditamap, uses an <anchorref> element to define a subset of the Astro-specific maintenance tasks. These tasks are attached to the general car prep procedure at run time, so that a complete, custom definition of the process of preparing an Astro for rental can be published to a hand-held device carried by the employee.

```
<map>
<!-- carprep.ditamap -->
 <topicref href="carPrep.dita">
   <topicref href="beforePrep.dita"/>
   <anchor id="prepDetail"/>
   <topicref href="afterPrep.dita"/>
 </topicref>
...
</map>

<map>
<!-- astro.ditamap -->
 <topicref href="astroTasks.dita">
   <topicref href="astroOverview.dita"/>
   <anchorref href="#prepDetail">
      <topicref href="astroChecklist.dita"/>
      <topicref href="otherPreparation.dita"/>
   </anchorref>
   <topicref href="astroConclusion.dita"/>
 </topicref>
</map>
```

If the @anchorref attribute on the <map> element in cartasks.ditamap is used to reference <anchor> element, then the entire map is integrated at that spot in carprep.ditamap:

```
<map>
<!-- carprep.ditamap -->
 <topicref href="carPrep.dita">
   <topicref href="beforePrep.dita"/>
   <anchor id="prepDetail"/>
   <topicref href="afterPrep.dita"/>
 </topicref>
...
</map>

<map anchorref="carprep.ditamap#prepDetail">
<!-- astro.ditamap -->
 <topicref href="astroTasks.dita">
   <topicref href="astroOverview.dita"/>
   <anchorref href="carprep.ditamap#prepDetail">
      <topicref href="astroChecklist.dita"/>
      <topicref href="otherPreparation.dita"/>
   </anchorref>
   <topicref href="astroConclusion.dita"/>
 </topicref>
</map>
```

Relationship tables

When a <topicgroup> element is placed in a relationship table, links are made to all topics referenced by the contained <topicref> elements.

However, when a <topicsetref> element is placed in a relationship table, just one link is generated. Consequently, a <topicset> element that is referenced from within a <relcell> element that is in a <relcell> element should have just one child element, with any additional topic references hierarchically subordinated under it. Depending on your reuse discipline, this may be a best practice for you to generalize to all instances of the <topicset> element.