Review K: Metadata and cascading metadata

Table of contents

1 DITA metadata

Metadata can be applied in both DITA topics and DITA maps. Metadata that is specified in DITA topics can be supplemented or overridden by metadata that is assigned in a DITA map. This design facilitates the reuse of DITA topics in different DITA maps and use-specific contexts.

DITA defines a core set of metadata elements to cover a variety of common scenarios. Because metadata requirements vary so widely, it is expected that few implementations will use the full range of these elements.

DITA also provides two generic elements, `<data>` and `<othermeta>`, which are intended for use when the core metadata elements do not provide the correct semantic. In addition, `<data>` is especially useful as a specialization base.

Requirements for rendering metadata vary widely. For that reason, any rendering of metadata in published content is left up to implementations.

1.1 Metadata elements

Metadata elements are available in both topics and DITA maps. This design enables authors and information architects to use identical metadata markup in both topics and maps.

When used in maps, metadata elements are located in the `<topicmeta>` element. When used in topics, metadata elements are located in the `<prolog>` element.

In general, specifying metadata in a `<topicmeta>` element that is a child of a `<topicref>` element is equivalent to specifying it in the `<prolog>` element of the referenced topic. The value of specifying the metadata in the map is that the topic then can be reused in other maps where different metadata might apply. Many items in the `<topicmeta>` element cascade to nested `<topicref>` elements within the map. See 2.2 Reconciling topic and map metadata elements (8) for information about which elements cascade.

1.2 Metadata attributes

Metadata attributes specify properties of the content that can be used to determine how the content is processed. Specialized metadata attributes can be defined to enable specific business-processing needs, such as semantic processing and data mining.

Metadata attributes typically are used for the following purposes:

- Filtering content based on the attribute values, for example, to suppress or publish profiled content
- Flagging content based on the attribute values, for example, to highlight specific content on output
- Performing custom processing, for example, to extract business-critical data and store it in a database

The base DITA vocabulary includes five specializations of the `@props` attribute as domains: `@audience`, `@deliveryTarget`, `@platform`, `@product`, and `@otherprops`. These five attributes are included in all the map and topic document-type shells that are provided with the specification.

Metadata attributes fall into the following categories.

Architectural attributes

The `@class`, `@DITAArchVersion`, and `@specializations` attributes provide metadata about the DITA source itself, such as what version of the grammar is used. These attributes are not intended for use in authored content.

Filtering and flagging attributes

The `@props` attribute and its specializations are intended for filtering. This includes the five specializations added to the OASIS document-type shells: `@audience`, `@deliveryTarget`, `@platform`, `@product`, and `@otherprops`.

These attributes plus the `@rev` attribute are intended for flagging.

Other metadata attributes

The `@status` and `@importance` attributes, many of the attributes available on the `<ux-window>` element, as well as custom attributes specialized from `@base`, are intended for application-specific behaviors. Such behaviors include aiding in search and retrieval, as well as controlling how a user assistance window is rendered.

Translation and localization attributes

The `@dir`, `@translate`, and `@xml:lang` attributes are intended for use with translating and localizing content.

1.3 Metadata in maps and topics

Metadata can be specified in both maps and topics. In most cases, metadata in the map either supplements or overrides metadata that is specified at the topic level.

Metadata can be specified by all the following mechanisms:

- Metadata elements that are located in the DITA map
- Specifying attributes on the `<map>` or `<topicref>` elements
- Metadata elements or attributes that are located in the DITA topic

Metadata elements and attributes in a map might apply to an individual topic, a set of topics, or globally for the entire document. Most metadata elements authored within a `<topicmeta>` element associate metadata with the parent element and its children. Because the topics in a branch of the hierarchy typically have some common subjects or properties, this is a convenient mechanism to define metadata for a set of topics.

Comment by rodaande on 8 Feb 2022
We should have a related link from this topic to the section on cascading; this is a conceptual topic about metadata and should not repeat the processing rules, but reading this I immediately want to know *which* elements cascade and how that works.

When the same metadata element or attribute is specified in both a map and a topic, by default the value in the map takes precedence. The assumption is that the map author has more knowledge of the reusing context than the topic author.

1.4 Window metadata for user assistance

Some user assistance topics might need to be displayed in a specific window or viewport, and this windowing metadata can be defined in the DITA map within the `<ux-window>` element.

In some help systems, a topic might need to be displayed in a window with a specific size or set of features. For example, a help topic might need to be displayed immediately adjacent to the user interface control that it supports in a window of a specific size that always remains on top, regardless of the focus within the operating system.

Application metadata that is specified on the <ux-window> element is closely tied to that specific application. It might be ignored when content is rendered for other uses.

Related reference

resourceid

ux-window

2 Metadata cascading

Metadata cascading is the process by which metadata elements and attributes specified for a map or for a topic reference cascade to nested references. This allows metadata properties to be set once and apply to an entire map or branch of a map.

2.1 Cascading of metadata attributes in a DITA map

Certain attributes cascade throughout a map, which facilitates attribute and metadata management. When attributes cascade, they apply to the elements that are children of the element where the attributes were specified. Cascading applies to a containment hierarchy, as opposed to a specialization hierarchy.

The following attributes cascade when set on the `<map>` element or when set within a map:

- `@rev`
- `@props` and any attribute specialized from `@props`, including those integrated by default in OASIS shells: `@audience`, `@deliveryTarget`, `@platform`, `@product`, `@otherprops`
- `@linking`, `@toc`, `@search`
- `@format`, `@scope`, `@type`
- `@xml:lang`, `@dir`, `@translate`
- `@processing-role`
- `@cascade`

Cascading is additive for attributes that accept multiple values, except when `cascade="nomerge"` is specified. For attributes that take a single value, the value that is defined on the closest containing element takes effect.

In a relationship table, metadata can be applied to entire rows or columns, as well as individual cells. The metadata cascade operates differently due to the nature of this tabular structure The cascade is not driven by a strict containment hierarchy because `<relcolspec>` elements do not contain child elements.

The following list illustrates how metadata cascades in a relationship table:

- `<reltable>`
  - `<relcolspec>`
    - `<relrow>`
      - `<relcell>`
        - `<topicref>`

Related reference

topicmeta

2.1.1 Processing cascading attributes in a map

Certain rules apply to processors when they process cascading attributes in a map.

001 (16)

When determining the value of an attribute, processors **MUST** evaluate each attribute on each individual element in a specific order. This order is specified in the following list. Applications **MUST** continue through the list until a value is established or until the end of the list is reached, at which point no value is

established for the attribute. In essence, the list provides instructions on how processors can construct a map where all attribute values are set and all cascading is complete.

002 (16) For attributes within a map, the following processing order **MUST** occur:

1. The `@conref` and `@keyref` attributes are evaluated.
2. The explicit values specified in the document instance are evaluated. For example, a `<topicref>` element with the `@toc` attribute set to "no" will use that value.
3. The default or fixed attribute values are evaluated. For example, the `@toc` attribute on the `<reltable>` element has a default value of "no".
4. The default values that are supplied by a controlled values file are evaluated.
5. The attributes cascade.
6. The processing-supplied default values are applied.
7. After the attributes are resolved within the map, *any values that do not come from processing-supplied defaults* will cascade to referenced maps.

   For example, most processors will supply a default value of `toc="yes"` when no `@toc` attribute is specified. However, a processor-supplied default of `toc="yes"` does not override a value of `toc="no"` that is set on a referenced map. If the `toc="yes"` value is explicitly specified, is given as a default through a DTD, RNG, or controlled values file, or cascades from a containing element in the map, it will override a `toc="no"` setting on the referenced map. See 2.3 Map-to-map cascading behaviors (10) for more details.
8. Repeat steps 1 (7) to 4 (7) for each referenced map.
9. The attributes cascade within each referenced map.
10. The processing-supplied default values are applied within each referenced map.
11. Repeat the process for maps referenced within the referenced maps.

For example, in the case of `<topicref toc="yes">`, applications must stop at item 2 (7) in the list; a value is specified for `@toc` in the document instance, so `@toc` values from containing elements will not cascade to that specific `<topicref>` element. The `toc="yes"` setting on that `<topicref>` element will cascade to contained elements, provided those elements reach item 5 (7) when evaluating the `@toc` attribute.

2.1.2 Merging of cascading attributes

The `@cascade` attribute can be used to modify the additive nature of attribute cascading, although it does not turn off cascading altogether. The attribute has two predefined values: "merge" and "nomerge".

merge
: Indicates that the metadata attributes cascade, and that the values of the metadata attributes are additive. This is the processing default for the `@cascade` attribute.

nomerge
: Indicates that the metadata attributes cascade, but that they are not additive for `<topicref>` elements that specify a different value for a specific metadata attribute. If the cascading value for an attribute is already merged based on multiple ancestor elements, that merged value continues to cascade until a new value is encountered. That is, setting `cascade="nomerge"` does not undo merging that took place on ancestor elements.

003 (16)

Implementers **MAY** define their own custom, implementation-specific tokens for the `@merge` attribute. To avoid name conflicts between implementations or with future additions to the standard, implementation-specific tokens **SHOULD** consist of a prefix that gives the name or an abbreviation for the implementation followed by a colon followed by the token or method name. For example, a processor might define the token "appToken:audience" in order to specify cascading and merging behaviors for **only** the `@audience` attribute.

004 (16)

The predefined values for the `@cascade` attribute **MUST** precede any implementation-specific tokens, for example, `cascade="merge appToken:audience"`.

2.2 Reconciling topic and map metadata elements

The `<topicmeta>` element in maps can contain numerous metadata elements. These metadata elements can have an effect on the parent `<topicref>` element, any child `<topicref>` elements, and – if a direct child of the `<map>` element – on the .

For each element that can be contained in the `<topicmeta>` element, the following table addresses the following questions:

How does it apply to the topic?

This column describes how the metadata specified within the `<topicmeta>` element interacts with the metadata specified in the referenced topic. In most cases, the properties are additive. For example, when a topic reference in a map contains `<category>installation</category>`, `<category>installation</category>` is added during processing to any metadata that is specified in the topic prolog.

Does it cascade to other topics in the map?

This column indicates whether the specified metadata element cascades to nested `<topicref>` elements. For example, when a topic reference in a map contains `<author>Jane Doe</author>`, `<author>Jane Doe</author>` is added during processing to the metadata for all child topic references. Some elements do not cascade.

What is the purpose when specified on the <map> element?

The map element permits metadata to be specified for the map. This column describes the effect that an element has when specified at this level.

When set on the <map> element, does it apply to all topics referenced in the map?

When specified on the `<map>` element element, some metadata elements then apply to all the topics that are referenced in the map.

Table 1: <topicmeta> elements and their properties

Element	How does it apply to the topic?	Does it cascade to child `<topicref>` elements?	What is the purpose when set on the `<map>` element?	When set on the `<map>` element, does it apply to all topics referenced in the map?
`<audience>`	Add to the topic	Yes	Specify an audience for the map	Yes
`<author>`	Add to the topic	Yes	Specify an author for the map	Yes

Element	How does it apply to the topic?	Does it cascade to child <topicref> elements?	What is the purpose when set on the <map> element?	When set on the <map> element, does it apply to all topics referenced in the map?
<category>	Add to the topic	Yes	Specify a category for the map	Yes
<copyright>	Add to the topic	Yes	Specify a copyright for the map	Yes
<critdates>	Add to the topic	Yes	Specify critical dates for the map	Yes
<data>	Add to the topic	No, unless specialized for a purpose that cascades	No stated purpose	No
<foreign>	Add to the topic	No, unless specialized for a purpose that cascades	No stated purpose	No
<keytext>	Not added to the topic	No	No stated purpose	No
<keywords>	Add to the topic	No	No stated purpose	No
<metadata>	Add to the topic	Yes	Specify metadata for the map	Yes
<othermeta>	Add to the topic	No	Define metadata for the map	Yes
<permissions>	Add to the topic	Yes	Specify permissions for the map	Yes
<prodinfo>	Add to the topic	Yes	Specify product info for the map	Yes
<publisher>	Add to the topic	Yes	Specify a publisher for the map	No
<resourceid>	Add to the topic	No	Specify a resource ID for the map itself	No
<shortdesc>	Applies only to links created based on this occurrence in the map	No	Provide a description of the map	No
<source>	Add to the topic	No	Specify a source for the map	No
<titlealt>	Add to the topic before its <titlealt> elements	No	Specify an alternative title for the map	No

Element	How does it apply to the topic?	Does it cascade to child <topicref> elements?	What is the purpose when set on the <map> element?	When set on the <map> element, does it apply to all topics referenced in the map?
<unknown>	Add to the topic	No, unless specialized for a purpose that cascades	No stated purpose	No
<ux-window>	Not added to the topic	No	Definitions are global, so setting at map level is equivalent to setting anywhere else.	No

Related reference

topicmeta

2.3 Map-to-map cascading behaviors

When a DITA map or map branch is referenced by another DITA map, by default certain rules apply. These rules pertain to the cascading behaviors of attributes, metadata elements, and the roles that are assigned to content , for example, the role of "Chapter" that is assigned by a <chapter> element. Attributes and elements that cascade within a map generally follow the same rules when cascading from one map to another map, but there are some exceptions and additional rules that apply.

2.3.1 Cascading of attributes from map to map

Certain attributes cascade from map to map.

The following attributes cascade from map to map:

- @rev
- @props and any attribute specialized from @props, including those integrated by default in OASIS shells: @audience, @deliveryTarget, @platform, @product, @otherprops
- @linking, @toc, @search
- @type
- @translate
- @processing-role
- @cascade

As with values that cascade within a map, the cascading is additive if the attribute permits multiple values, such as @audience. For attributes that take a single value, the value that is defined on the closest containing element takes effect.

The following attributes do not cascade from map to map

@format

The @format attribute is set to "ditamap" when a map or map branch is referenced, so it cannot cascade through to the referenced map.

@scope

The value of the @scope attribute describes the map itself, rather than the content. For example, when the @scope attribute is set to "external", it indicates that the referenced map itself is external and unavailable, so the value cannot cascade into that referenced map.

@xml:lang and @dir

Cascading behavior for `@xml:lang` is defined in The xml:lang attribute. The `@dir` attribute follows the same rules as `@xml:lang`.

While the `@class` attribute is unique and does not cascade, the value of the attribute is used to determine the processing roles that cascade from map to map. See 2.3.3 Cascading of roles from map to map (11) for more information.

2.3.2 Cascading of metadata elements from map to map

Elements that are contained within `<topicmeta>` elements follow the same rules for cascading from map to map as the rules that apply within a single DITA map.

For a complete list of which elements cascade within a map, see the column "Does it cascade to child `<topicref>` elements?" in the topic 2.2 Reconciling topic and map metadata elements (8).

Note It is possible that a specialization might define metadata that is intended to replace rather than add to metadata in the referenced map, but DITA, by default, does not have a mechanism to specify this behavior.

2.3.3 Cascading of roles from map to map

When specialized `<topicref>` elements, such as `<chapter>` or `<mapref>`, reference a map, they typically imply a semantic role for the referenced content.

Comment by Kristen J Eberlein on 04 July 2019

We need to look at the instances of "should" in this topic. Can they be recast? Do we need to introduce RFC-2119 language?

The semantic role reflects the `@class` hierarchy of the referencing `<topicref>` element. It is equivalent to having the `@class` attribute from the referencing `<topicref>` cascade to the top-level `<topicref>` elements in the referenced map. Although this cascade behavior is not universal, there are general guidelines for when a role based on the `@class` attribute cascades.

When a `<topicref>` element or a specialization of a `<topicref>` element references a DITA resource, it defines a role for that resource. In some cases this role is straightforward, such as when a `<topicref>` element references a DITA topic (giving it the already known role of "topic"), or when a `<mapref>` element references a DITA map (giving it the role of "DITA map").

Comment by rodaande on 8 Feb 2022

The following paragraph includes the statement: "the non-default behavior should be clearly specified"

We do not say how or where. For mapgroup, I believe it is only specified here in this topic. I think either we need this as part of every mapgroup element definition, in "Processing expectations", or we need a clear table here listing every element where this behavior does not apply.

Unless otherwise instructed, a specialized `<topicref>` element that references a map supplies a role for the referenced content. This means that, in effect, the `@class` attribute of the referencing element cascades to top-level topicref elements in the referenced map. In situations where this should not happen —such as all elements from the mapgroup domain—the non-default behavior should be clearly specified.

For example, when a `<chapter>` element from the bookmap specialization references a map, it supplies a role of "chapter" for each top-level `<topicref>` element in the referenced map. When the `<chapter>` element references a branch in another map, it supplies a role of "chapter" for that branch. In effect, the

`@class` attribute for `<chapter>` (`"- map/topicref bookmap/chapter "`) cascades to the top-level `<topicref>` elements in the nested map, although it does not cascade any further.

005 (16) Because the `<mapref>` element is a convenience element, the top-level `<topicref>` elements in the map referenced by a `<mapref>` element **MUST NOT** be processed as if they are `<mapref>` elements. The `@class` attribute from the `<mapref>` element (`"+ map/topicref mapgroup-d/mapref "`) does not cascade to the referenced map.

006 (16) In some cases, preserving the role of the referencing element might result in out-of-context content. For example, a `<chapter>` element that references a bookmap might pull in `<part>` elements that contain nested `<chapter>` elements. Treating the `<part>` element as a `<chapter>` will result in a chapter that nests other chapters, which is not valid in bookmap and might not be understandable by processors. The result is implementation specific. Processors **MAY** choose to treat this as an error, issue a warning, or simply assign new roles to the problematic elements.

2.4 Examples of metadata cascading

These examples illustrate the processing expectations for cascading metadata. The processing examples use either before and after sample markup or expanded syntax that shows the equivalent markup withough cascading.

2.4.1 Example: How map-level metadata elements cascade to the referenced topics

In this scenario, elements located in the`<topicmeta>` element for a map cascade to the referenced topics.

The following code sample illustrates how an information architect can apply certain metadata to all the DITA topics in a map:

```
<map xml:lang="en-us">
  <title>DITA maps</title>
  <topicmeta>
    <author>Kristen James Eberlein</author>
    <copyright>
      <copyryear year="2020"/>
    <copyrholder>OASIS</copyrholder>
    </copyright>
  </topicmeta>
  <topicref href="dita-maps.dita">
    <topicref href="definition_ditamaps.dita"/>
    <topicref href="purpose_ditamaps.dita"/>
    <!-- ... -->
  </topicref>
</map>
```

The author and copyright information cascades to each of the DITA topics that are referenced in the DITA map. When the DITA map is processed to HTML5, for example, the author and copyright metadata apply to each generated HTML5 file.

2.4.2 Example: How metadata elements cascade from one map to another

In this scenario, a metadata element that is located in a map reference cascades to the topics that are referenced in a nested map.

Consider the following code examples:

Figure 1: Root map

```
<map>
  <title>Acme User Guide</title>
  <topicref href="acme-defects.ditamap" format="ditamap">
    <topicmeta>
      <shortdesc>This map contains information about Acme defects.</shortdesc>
    </topicmeta>
  </topicref>
  <topicref href="installing.ditamap" format="ditamap">
    <topicmeta>
      <audience type="installer"/>
    </topicmeta>
  </topicref>
  <mapref href="troubleshooting.ditamap"/>
  <mapref href="reference.ditamap"/>
</map>
```

Figure 2: installing.ditamap

```
<map>
  <title>Installation topics</title>
  <topicmeta>
    <audience type="administrator"/>
  </topicmeta>
  <topicref href="install-1.dita"/>
  <topicref href="install-2.dita"/>
</map>
```

When the root map is processed, the following behavior occurs:

- Because the `<shortdesc>` element does not cascade, it does not apply to the DITA topics that are referenced in `acme-defects.ditamap`.
- Because the `<audience>` element cascades, the `<audience>` element in the reference to `installing.ditamap` combines with the `<audience>` element that is specified at the top level of `installing.ditamap`. The result is that the `install-1.dita` and `install-2.dita` topics are processed as though they each contained the following child `<topicmeta>` element:

  ```
  <topicmeta>
    <audience type="installer"/>
    <audience type="administrator"/>
  </topicmeta>
  ```

2.4.3 Example: How attributes cascade from one map to another

In this scenario, attributes in one map cascade to a nested map.

Assume the following references in `test.ditamap`:

```
<map>
  <topicref href="a.ditamap" format="ditamap" toc="no"/>
  <mapref   href="b.ditamap" audience="developer"/>
  <mapref   href="c.ditamap#branch2" platform="myPlatform"/>
</map>
```

- The map `a.ditamap` is treated as if `toc="no"` is specified on the root `<map>` element. This means that the topics that are referenced by `a.ditamap` do not appear in the navigation generated by `test.ditamap`, except for branches within the map that explicitly set `toc="yes"`.
- The map `b.ditamap` is treated as if `audience="developer"` is set on the root `<map>` element. If the `@audience` attribute is already set on the root `<map>` element within `b.ditamap`, the value "developer" is added to any existing values.
- The element with `id="branch2"` within the map `c.ditamap` is treated as if `platform="myPlatform"` is specified on that element. If the `@platform` attribute is already specified on the element with `id="branch"`, the value"myPlatform" is added to existing values.

2.4.4 Example: How the @cascade attribute affects attribute cascading

In this scenario, the `@cascade` attribute is used to modify how metadata attributes cascade within a map.

Figure 3: Example of cascade="merge"

Consider the following code example:

```
<map audience="a b" cascade="merge">
     <topicref href="topic.dita" audience="c"/>
</map>
```

In this map, the `cascade="merge"` attribute instructs a processor to merge attribute values while cascading. With `@audience` specified on both the `<map>` element and the `<topicref>` element, the effective `@audience` attribute value for the reference to `topic.dita` is "a b c".

Figure 4: Example of cascade="nomerge"

Consider the following code example:

```
<map audience="a b" cascade="nomerge">
     <topicref href="topic.dita" audience="c"/>
</map>
```

In this map, the `cascade="nomerge"` attribute instructs a processor *not* to merge attribute values while cascading. With `@audience` specified on both the `<map>` element and the `<topicref>` element, the effective `@audience` attribute value on the reference to `topic.dita` is not merged with the value from the map and remains "c".

Figure 5: Example of changing the @cascade value within the map

Consider the following code example:

```
<map platform="a" product="x" cascade="merge">
  <topicref href="one.dita" platform="b" product="y">
    <topicref href="two.dita" cascade="nomerge" product="z"/>
  </topicref>
</map>
```

In this map, the `@cascade` attribute is set to "merge" at the map level but changes to "nomerge" on a topic reference.

- For the topic reference to `one.dita`, `cascade="merge"` is specified. This results in an effective `@platform` value of "a b" and an effective `@product` value of "x y".
- The topic reference to `two.dita` specifies `cascade="nomerge"`, so attribute values from other elements do not merge with anything specified on the topic reference. The `@platform` attribute is

not specified, so the effective value is "a b", which still cascades from the parent element. The `@product` value does not merge with values from the parent, so the effective value is "z".

2.4.5 Example: How <topicref> roles cascade to referenced maps

In this scenario, a specialized `<topicref>` element references content in another map.

Consider the scenario of a `<chapter>` element from the Book map specialization that references a DITA map. This scenario could take several forms:

Referenced map contains a single top-level <topicref> element
The entire branch functions as if it were included in the bookmap. The top-level `<topicref>` element is processed as if it were the `<chapter>` element.

Referenced map contains multiple top-level <topicref> elements
Each top-level `<topicref>` element is processed as if it were a `<chapter>` element, since the processing role of the `<chapter>` element cascades.

Referenced map contains a single <appendix> element
The `<appendix>` element is processed as it were a `<chapter>` element.

Referenced map contains a single <part> element, with nested <chapter> elements
The `<part>` element is processed as it were a `<chapter>` element. Nested `<chapter>` elements might not be understandable by processors, although applications can recover as described above.

<chapter> element references a single <topicref> element rather than a map
The referenced `<topicref>` element is processed as if it were a `<chapter>` element.

A Aggregated RFC-2119 statements

This appendix contains all the normative statements from the DITA 2.0 specification. They are aggregated here for convenience in this non-normative appendix.

Item	Conformance statement
001 (6)	When determining the value of an attribute, processors **MUST** evaluate each attribute on each individual element in a specific order. This order is specified in the following list. Applications **MUST** continue through the list until a value is established or until the end of the list is reached, at which point no value is established for the attribute. In essence, the list provides instructions on how processors can construct a map where all attribute values are set and all cascading is complete.
002 (7)	For attributes within a map, the following processing order **MUST** occur: **1.** The `@conref` and `@keyref` attributes are evaluated. **2.** The explicit values specified in the document instance are evaluated. For example, a `<topicref>` element with the `@toc` attribute set to "no" will use that value. **3.** The default or fixed attribute values are evaluated. For example, the `@toc` attribute on the `<reltable>` element has a default value of "no". **4.** The default values that are supplied by a controlled values file are evaluated. **5.** The attributes cascade. **6.** The processing-supplied default values are applied. **7.** After the attributes are resolved within the map, *any values that do not come from processing-supplied defaults* will cascade to referenced maps. For example, most processors will supply a default value of `toc="yes"` when no `@toc` attribute is specified. However, a processor-supplied default of `toc="yes"` does not override a value of `toc="no"` that is set on a referenced map. If the `toc="yes"` value is explicitly specified, is given as a default through a DTD, RNG, or controlled values file, or cascades from a containing element in the map, it will override a `toc="no"` setting on the referenced map. See 2.3 Map-to-map cascading behaviors (10) for more details. **8.** Repeat steps 1 to 4 for each referenced map. **9.** The attributes cascade within each referenced map. **10.** The processing-supplied default values are applied within each referenced map. **11.** Repeat the process for maps referenced within the referenced maps.
003 (8)	Implementers **MAY** define their own custom, implementation-specific tokens for the `@merge` attribute. To avoid name conflicts between implementations or with future additions to the standard, implementation-specific tokens **SHOULD** consist of a prefix that gives the name or an abbreviation for the implementation followed by a colon followed by the token or method name. For example, a processor might define the token "appToken:audience" in order to specify cascading and merging behaviors for **only** the `@audience` attribute.
004 (8)	The predefined values for the `@cascade` attribute **MUST** precede any implementation-specific tokens, for example, `cascade="merge appToken:audience"`.
005 (12)	Because the `<mapref>` element is a convenience element, the top-level `<topicref>` elements in the map referenced by a `<mapref>` element **MUST NOT** be processed as if they are `<mapref>` elements. The `@class` attribute from the `<mapref>` element (`"+ map/topicref mapgroup-d/mapref "`) does not cascade to the referenced map.
006 (12)	In some cases, preserving the role of the referencing element might result in out-of-context content. For example, a `<chapter>` element that references a bookmap might pull in `<part>` elements that contain nested `<chapter>` elements. Treating the `<part>` element as a `<chapter>` will result in a chapter

Item	Conformance statement
	that nests other chapters, which is not valid in bookmap and might not be understandable by processors. The result is implementation specific. Processors **MAY** choose to treat this as an error, issue a warning, or simply assign new roles to the problematic elements.

Index

C

D

E

M

T