Stage 3 proposal: Feature #253 Indexing changes

Refactor indexing to remove redundant elements and reduce complexity

Champion

Kristen James Eberlein

Tracking information

Event	Date	Links
Stage 1 proposal accepted	11 June 2019	11 June 2019
Stage 2 proposal submitted	14 June 2019	DITA PDF
Stage 2 proposal discussed	18 June 2019	18 June 2019
Stage 2 proposal approved	02 July 2019	Include link to meeting minutes
Stage 3 proposal submitted to reviewers	02 July 2019	Bill Burns Eliot Kimber Dawn Stevens
Stage 3 proposal (this document) submitted to TC		

Approved technical requirements

- Remove the indexing domain, and add `<index-see>` and `<index-see-also>` to the base
- Remove `<index-base>`
- Remove `<index-sort-as>`

Dependencies or interrelated proposals

None

Modified grammar files

The following files must be modified:

DTDs

- `basemap.dtd`
- `basetopic.dtd`
- `commonElementsMod.dtd`
- `commonElementsMod.ent`

RNG

- `basemap.rng`
- `basetopic.rng`

- `commonElementsMod.rng`

In the content below, the following conventions are used:

- Bold is used to indicate code to be added, for example, **addition**.
- Line-through is used to indicate code to be removed, for example, ~~removal~~.
- Ellipses (…) indicate where code is snipped for brevity.

Figure 1: Changes to basemap.dtd

```
<!-- ============================================================= -->
<!--              DOMAIN ENTITY DECLARATIONS                       -->
<!-- ============================================================= -->
...
<!ENTITY % indexing-d-dec
  PUBLIC "-//OASIS//ENTITIES DITA 1.3 Indexing Domain//EN"
         "indexingDomain.ent"
>%indexing-d-dec;
...
<!-- ============================================================= -->
<!--                     DOMAIN EXTENSIONS                         -->
<!-- ============================================================= -->
<!--                     One for each extended base element, with
                        the name of the domain(s) in which the
                        extension was declared                    -->
...
<!ENTITY % index-base    "index-base |
                          %indexing-d-index-base;
                         ">
...
<!-- ============================================================= -->
<!--                     DOMAINS ATTRIBUTE OVERRIDE                -->
<!-- ============================================================= -->

<!ENTITY included-domains
                          "&mapgroup-d-att;
                           &delay-d-att;
                           &deliveryTargetAtt-d-att;
                           &ditavalref-d-att;
                           &indexing-d-att;
                           &hazard-d-att;
                           &hi-d-att;
                           &ut-d-att;
  "
>
...
<!-- ============================================================= -->
<!--                     DOMAIN ELEMENT INTEGRATION                -->
<!-- ============================================================= -->
...

<!ENTITY % indexing-d-def
  PUBLIC "-//OASIS//ELEMENTS DITA 1.3 Indexing Domain//EN"
         "indexingDomain.mod"
>%indexing-d-def;
```

Figure 2: Changes to basetopic.dtd

```
<!-- ============================================================= -->
<!--              DOMAIN ENTITY DECLARATIONS                       -->
<!-- ============================================================= -->
...
<!ENTITY % indexing-d-dec
  PUBLIC "-//OASIS//ENTITIES DITA 1.3 Indexing Domain//EN"
         "indexingDomain.ent"
>%indexing-d-dec;
...
<!-- ============================================================= -->
<!--                     DOMAIN EXTENSIONS                         -->
<!-- ============================================================= -->
```

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

<!ENTITY % index-base    "index-base |
                          %indexing-d-index-base;
                         ">
...
<!-- ============================================================= -->
<!--                    DOMAINS ATTRIBUTE OVERRIDE                 -->
<!-- ============================================================= -->

<!ENTITY included-domains
                           "&deliveryTargetAtt-d-att;
                            &hazard-d-att;
                            &hi-d-att;
                            &indexing-d-att;
                            &ut-d-att;
  "
>
...
<!-- ============================================================= -->
<!--                    DOMAIN ELEMENT INTEGRATION                 -->
<!-- ============================================================= -->
...
<!ENTITY % indexing-d-def
  PUBLIC "-//OASIS//ELEMENTS DITA 1.3 Indexing Domain//EN"
         "indexingDomain.mod"
>%indexing-d-def;
...
```

Figure 3: Changes to commonElements.ent

```
...
<!-- ============================================================= -->
<!--                    ELEMENT NAME ENTITIES                      -->
<!-- ============================================================= -->
...
<!ENTITY % index-base  "index-base"                                 >
...
```

Figure 4: Changes to commonElements.mod

```
...
<!--                    LONG NAME: Index Term                      -->
<!ENTITY % indexterm.content
                        "(%words.cnt; |
                          %ph; |
                          %indexterm; |
                          %index-see;
                          %index-see-also;
                          %index-base;)*"
>
...
<!--                    LONG NAME: Index Base                      -->
<!ENTITY % index-base.content
                        "(%words.cnt; |
                          %indexterm;)*"
>
<!ENTITY % index-base.attributes
             "keyref
                         CDATA
                                    #IMPLIED
              %univ-atts;"
>
<!ELEMENT  index-base %index-base.content;>
<!ATTLIST  index-base %index-base.attributes;>
<!--                    LONG NAME: Index See                       -->
<!ENTITY % index-see.content
                        "(%words.cnt; |
                          %ph; |
```

```
                        %indexterm;)*"
>
<!ENTITY % index-see.attributes
             "keyref
                         CDATA
                                    #IMPLIED
              %univ-atts;"
>
<!ELEMENT  index-see %index-see.content;>
<!ATTLIST  index-see %index-see.attributes;>


<!--                    LONG NAME: Index See Also                  -->
<!ENTITY % index-see-also.content
                       "(%words.cnt; |
                         %ph; |
                         %indexterm;)*"
>
<!ENTITY % index-see-also.attributes
             "keyref
                         CDATA
                                    #IMPLIED
              %univ-atts;"
>
<!ELEMENT  index-see-also %index-see-also.content;>
<!ATTLIST  index-see-also %index-see-also.attributes;>
...
<!-- ============================================================= -->
<!--             SPECIALIZATION ATTRIBUTE DECLARATIONS             -->
<!-- ============================================================= -->
...
<!ATTLIST  index-base    %global-atts;  class CDATA "- topic/index-base " >
<!ATTLIST  index-see    %global-atts;  class CDATA "- topic/index-see ">
<!ATTLIST  index-see-also %global-atts;  class CDATA "- topic/index-see-also ">
...
```

Figure 5: Changes to basemap.rng

```
...
  <div>
      <a:documentation>DITA DOMAINS ATTRIBUTE</a:documentation>

      <define name="domains-att">
         <optional>
            <attribute name="domains"
                       a:defaultValue="
                         (topic delay-d)
                         (map ditavalref-d)
                         (topic hazard-d)
                         (topic hi-d)
                         (topic indexing-d)
                         (map mapgroup-d)
                         (topic ut-d)
                         a(props deliveryTarget)"/>
         </optional>
      </define>


  </div>
  <div>
      <a:documentation>MODULE INCLUSIONS</a:documentation>

      <include href="mapMod.rng"/>
      <include href="mapGroupDomain.rng"/>

      <include href="delayResolutionDomain.rng"/>
      <include href="deliveryTargetAttDomain.rng" dita:since="1.3"/>
      <include href="ditavalrefDomain.rng" dita:since="1.3"/>
      <include href="indexingDomain.rng"/>
      <include href="hazardstatementDomain.rng"/>
      <include href="highlightDomain.rng"/>
      <include href="utilitiesDomain.rng"/>
```

```
  </div>
...
```

Figure 6: Changes to basetopic.rng

```
...
  <div>
    <a:documentation>DOMAINS ATTRIBUTE</a:documentation>
    <define name="domains-att">
      <optional>
        <attribute name="domains"
          a:defaultValue="(topic hazard-d)
                          (topic hi-d)
                          (topic indexing-d)
                          (topic ut-d)
                          a(props deliveryTarget)"
        />
      </optional>
    </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="deliveryTargetAttDomain.rng" dita:since="1.3"/>
    <include href="hazardstatementDomain.rng" dita:since="1.3"/>
    <include href="highlightDomain.rng"/>
    <include href="indexingDomain.rng"/>
    <include href="utilitiesDomain.rng"/>
  </div>
...
```

Figure 7: Changes to commonElementsMod.rng

```
...
    <define name="index-base">
      <ref name="index-base.element"/>
    </define>
    <define name="index-see">
      <ref name="index-see.element"/>
    </define>
    <define name="index-see-also">
      <ref name="index-see-also.element"/>
    </define>
...
<div>
        <a:documentation>LONG NAME: Index Term</a:documentation>
        <define name="indexterm.content">
          <zeroOrMore>
            <choice>
              <ref name="words.cnt"/>
              <ref name="ph" dita:since="1.3"/>
              <ref name="indexterm"/>
              <ref name="index-base"/>
              <ref name="index-see"/>
              <ref name="index-see-also"/>
            </choice>
          </zeroOrMore>
        </define>
        <define name="indexterm.attributes">
          <optional>
            <attribute name="keyref"/>
          </optional>
          <optional>
            <attribute name="start"/>
          </optional>
          <optional>
```

```
            <attribute name="end"/>
          </optional>
          <ref name="univ-atts"/>
        </define>
        <define name="indexterm.element">
          <element name="indexterm" dita:longName="Index Term">
            <a:documentation>An &lt;indexterm> element allows the author to indicate that a
certain word or phrase should produce an index entry in the generated index. Category:
Miscellaneous
              elements</a:documentation>
            <ref name="indexterm.attlist"/>
            <ref name="indexterm.content"/>
          </element>
        </define>
        <define name="indexterm.attlist" combine="interleave">
          <ref name="indexterm.attributes"/>
        </define>
      </div>
   <div>
      <a:documentation>LONG NAME: Index See</a:documentation>
      <define name="index-see.content">
        <zeroOrMore>
          <choice>
            <ref name="words.cnt"/>
            <ref name="ph" dita:since="1.3"/>
            <ref name="indexterm"/>
          </choice>
        </zeroOrMore>
      </define>
      <define name="index-see.attributes">
        <optional>
          <attribute name="keyref"/>
        </optional>
        <ref name="univ-atts"/>
      </define>
      <define name="index-see.element">
        <element name="index-see" dita:longName="Index See">
          <a:documentation>An &lt;index-see> element within an &lt;indexterm> directs the
reader to another index entry that the reader should reference instead of the current one.
          </a:documentation>
          <ref name="index-see.attlist"/>
          <ref name="index-see.content"/>
        </element>
      </define>
      <define name="index-see.attlist" combine="interleave">
        <ref name="index-see.attributes"/>
      </define>
    </div>
    <div>
      <a:documentation>LONG NAME: Index See Also</a:documentation>
      <define name="index-see-also.content">
        <zeroOrMore>
          <choice>
            <ref name="words.cnt"/>
            <ref name="ph" dita:since="1.3"/>
            <ref name="indexterm"/>
          </choice>
        </zeroOrMore>
      </define>
      <define name="index-see-also.attributes">
        <optional>
          <attribute name="keyref"/>
        </optional>
        <ref name="univ-atts"/>
      </define>
      <define name="index-see-also.element">
        <element name="index-see-also" dita:longName="Index See Also">
          <ref name="index-see-also.attlist"/>
          <ref name="index-see-also.content"/>
        </element>
      </define>
      <define name="index-see-also.attlist" combine="interleave">
        <ref name="index-see-also.attributes"/>
      </define>
```

```
    </div>
...
      <div>
        <a:documentation>LONG NAME: Index Base</a:documentation>
        <define name="index-base.content">
          <zeroOrMore>
            <choice>
              <ref name="words.cnt"/>
              <ref name="indexterm"/>
            </choice>
          </zeroOrMore>
        </define>
        <define name="index-base.attributes">
          <optional>
            <attribute name="keyref"/>
          </optional>
          <ref name="univ-atts"/>
        </define>
        <define name="index-base.element">
          <element name="index-base" dita:longName="Index Base">
            <a:documentation>The &lt;index-base> element allows indexing extensions to be
added by specializing off this element. It does not in itself have any meaning and should be
ignored in
              processing. Category: Miscellaneous elements</a:documentation>
            <ref name="index-base.attlist"/>
            <ref name="index-base.content"/>
          </element>
        </define>
        <define name="index-base.attlist" combine="interleave">
          <ref name="index-base.attributes"/>
        </define>
-
      </div>
...
  <div>
    <a:documentation>SPECIALIZATION ATTRIBUTE DECLARATIONS</a:documentation>
...
    <define name="index-base.attlist" combine="interleave">
      <ref name="global-atts"/>
      <optional>
        <attribute name="class" a:defaultValue="- topic/index-base "/>
      </optional>
    </define>

        <define name="index-see.attlist" combine="interleave">
      <ref name="global-atts"/>
      <optional>
        <attribute name="class" a:defaultValue="- topic/index-see "/>
      </optional>
    </define>
    <define name="index-see-also.attlist" combine="interleave">
      <ref name="global-atts"/>
      <optional>
        <attribute name="class" a:defaultValue="- topic/index-see-also "/>
      </optional>
    </define>
```

Modified terminology

None

Modified specification documentation

Note All numeric references here are to the DITA 2.0 specification, working draft 04.

The following topic cluster needs to be modified and relocated:

- 9.5.4 Indexing-group domain elements
  - 9.5.4.1 `<indexterm>`

- 9.5.4.2 <index-see>
- 9.5.4.3 <index-see-also>
- ~~9.5.4.4 <index-sort-as>~~

The topic cluster should be re-named "Indexing elements"; the <index-sort-as> topic removed; and the cluster relocated to be a child of "9.2 "Body elements".

The cross reference to the <index-sort-as> topic should be removed from "9.1 DITA elements, A to Z."

The relationship-table link from 6.8 "Sorting" to 9.5.44.4 <index-sort-as> should be removed.

Any index entries to <index-sort-as> need to be removed.

Content from 9.5.44.4 <index-sort-as> is either incorporated into a new Index sorting (10) topic or moved into 9.5.6.5 <sort-as>.

The following table contains precise suggestions for changes to be made to other topics. The following conventions are used to indicate textual changes:

- Deletions are indicated with line through and red text, for example, ~~deletion~~.
- Insertions are indicated with underlining and green text, for example, insertion.

Topic	DITA 2.0, working draft 04 content	New content
6 DITA processing	DITA processing is affected by a number of factors, including attributes that indicate the set of vocabulary and constraint modules on which a DITA document depends; navigation; linking; content reuse (using direct or indirect addressing); conditional processing; branch filtering; chunking; and more. In addition, translation of DITA content is expedited through the use of the @dir, @translate, and @xml:lang attributes, and the <index-sort-as> element.	DITA processing is affected by a number of factors, including attributes that indicate the set of vocabulary and constraint modules on which a DITA document depends; navigation; linking; content reuse (using direct or indirect addressing); conditional processing; branch filtering; chunking; and more. In addition, translation of DITA content is expedited through the use of the @dir, @translate, and @xml:lang attributes~~, and the <index-sort-as> element~~.
6.6 Translation and localization	DITA has features that facilitate preparing content for translation and working with multilingual content, including the @xml:lang attribute, the @dir attribute, and the @translate attribute. In addition, the <sort-as> and <index-sort-as> elements provide support for sorting in languages in which the correct sorting of an element requires text that is different from the base content of the element.	DITA has features that facilitate preparing content for translation and working with multilingual content, including the @xml:lang attribute, the @dir attribute, and the @translate attribute. In addition, the <sort-as>~~ and <index-sort-as> elements provide~~ element provides support for sorting in languages in which the correct sorting of an element requires text that is different from the base content of the element.
6.8 Sorting	See the Sorting (11) topic below.	See the modified Sorting (11) topic below. I want TC members to view the changes within the context of the entire topic.

Topic	DITA 2.0, working draft 04 content	New content
9.5.6.5 `<sort-as>`	See the sort-as (12) topic below.	See the modified sort-as (12) topic below. I want TC members to view the changes within the context of the entire topic.

Migration plans for backwards incompatibilities

See the "Migration plan" section of the stage two proposal.

Index sorting

~~The <index-sort-as> element~~ The combination of an <indexterm> and <sort-as> element specifies a sort phrase under which an index entry ~~would be~~ is sorted.

This ~~element~~ gives an author the flexibility to sort an index entry in an index differently from how its text normally would be sorted. The common use for this is to disregard insignificant leading text, such as punctuation or words like "the" or "a". For example, the author might want <data> to be sorted under the letter D rather than the left angle bracket (<). An author might want to include such an entry under both the punctuation heading and the letter D, in which case there can be two index entry directives differentiated only by the sort order.

Certain languages have special sort order needs. For example, Japanese index entries might be written partially or wholly in kanji, but need to be sorted in phonetic order according to its hiragana/katakana rendition. There is no reliable automated way to map written to phonetic text: for kanji text, there can be multiple phonetic possibilities depending on the context. The only way to correctly sort Japanese index entries is to keep the phonetic counterparts with the written forms. The phonetic text would be presented as the sort order text for indexing purposes.

Sorting

Processors can be configured to sort elements. Typical processing includes sorting glossary entries, index entries, lists of parameters or reference entries in custom navigation structures, and tables based on the contents of cells in specific columns or rows.

Each element to be sorted must have some inherent text on which it will be sorted. This text is the *base sort phrase* for the element. For elements that have titles, the base sort phrase usually is the content of the `<title>` element. For elements that do not have titles, the base sort phrase might be literal content in the DITA source, or it might be generated or constructed based on the semantics of the element involved; for example, it could be constructed from various attribute or metadata values.

Processors that perform sorting **SHOULD** explicitly document how the base sort phrase is determined for a given element.

The `<sort-as>` element can be used to specify an effective sort phrase when the base sort phrase is not appropriate for sorting. For index terms, the ~~`<index-sort-as>`~~ `<sort-as>` element ~~can be used to specify~~ specifies the effective sort phrase for an index entry.

The details of sorting and grouping are implementation specific. Processors might provide different mechanisms for defining or configuring collation and grouping details. Even where the `<sort-as>` element is specified, two processors might produce different sorted and grouped results because they might use different collation and grouping rules. For example, one processor might be configured to sort English terms before non-English terms, while another might be configured to sort them after. The grouping and sorting of content is subject to local editorial rules.

When a `<sort-as>` element is specified, processors that sort the containing element **MUST** construct the effective sort phrase by prepending the content of the `<sort-as>` element to the base sort phrase. This ensures that two items with the same `<sort-as>` element but different base sort phrases will sort in the appropriate order.

For example, if a processor uses the content of the `<title>` element as the base sort phrase, and the title of a topic is "24 Hour Support Hotline" and the value of the `<sort-as>` element is "twenty-four hour", then the effective sort phrase would be "twenty-four hour24 Hour Support Hotline".

<sort-as>

For elements that are sorted, the <sort-as element> provides text that is combined with the base sort phrase to construct the effective sort phrase. The text can be specified in the content of the <sort-as element> or in the value attribute on the <sort-as element> element. The <sort-as element> element also is useful for elements where the base sort phrase is inadequate or non-existent, for example, a glossary or index entry for a Japanese Kanji phrase.

Usage information

The <sort-as> element can contain <text> and <keyword> elements in order to enable content referencing. If a <keyword> element is used within <sort-as>, the @keyref attribute can be used to set the sort phrase. If a <keyword> uses @keyref and would otherwise also act as a navigation link, the link aspect of the @keyref attribute is ignored.

Some elements in the base DITA vocabulary are natural candidates for sorting, including topics, definition list entries, index entries,and rows in tables and simple tables. Authors are likely to include <sort-as> elements in the following locations:

- For topics, the <sort-as> element can be included directly in <title>, <searchtitle>, or <navtitle> when the different forms of title need different effective sort phrases. If the effective sort phrase is common to all the titles for a topic, the <sort-as> element can be included as a direct child of the topic prolog anywhere <data> is allowed.
- For glossary entry topics, the <sort-as> element can be included directly in <glossterm> or as a direct child of <prolog>.
- For topic references, the <sort-as> element can be included directly in the <navtitle> or <title> element within <topicmeta> or as a child of <topicmeta>.
- For definition list items, include the <sort-as> element in the <dt> element.
- For index entries, the <sort-as> can be included as a child of <indexterm>. In a multilevel index entry, the <sort-as> element only affects the level in which it occurs.

Processing expectations

As a specialization of <data>, the <sort-as> element is allowed in any context where <data> is allowed. However, the presence of <sort-as> within an element does not, by itself, indicate that the containing element should be sorted. Processors can choose to sort any DITA elements for any reason. Likewise, processors are not required to sort any elements. See [sorting topic] for more information on sorting.

Processors **SHOULD** expect to encounter <sort-as> elements in the above locations. Processors that sort **SHOULD** use the following precedence rules:

- A <sort-as> element that is specified in a title takes precedence over a <sort-as> element that is specified as a child of the topic prolog.
- Except for instances in the topic prolog, processors only apply <sort-as> elements that are either a direct child of the element to be sorted or a direct child of the title- or label-defining element of the element to be sorted.
- When an element contains multiple, direct-child, <sort-as> elements, the first direct-child <sort-as> element in document order takes precedence.

- ~~When located within the <indexterm> element, the <sort-as> element is equivalent to <index-sort-as>. It is an error for an <indexterm> element to directly contain both <sort-as> and <index-sort-as> elements.~~
- It is an error if there is more than one <sort-as> child for a given <indexterm>. An implementation might give an error message.
- Sort phrases are determined after filtering and content reference resolution occur.

When a <sort-as> element is specified, processors that sort the containing element **MUST** construct the effective sort phrase by prepending the content of the <sort-as> element to the base sort phrase. This ensures that two items with the same <sort-as> element but different base sort phrases will sort in the appropriate order.

For example, if a processor uses the content of the <title> element as the base sort phrase, and the title of a topic is "24 Hour Support Hotline" and the value of the <sort-as> element is "twenty-four hour", then the effective sort phrase would be "twenty-four hour24 Hour Support Hotline".

Specialization hierarchy

The <sort-as> element is specialized from <data>. It in defined in the utilities-domain module.

Attributes

The following attributes are available on this element: and the attributes defined below.

@name
Names the metadata item that the element represents. The default value is "sort-as". Specializations of <sort-as> can set the default value of the @name attribute to reflect the tag name of the specialized element.

@value
The value of the metadata item. When the <sort-as> element has content and the @value attribute is specified, the @value attribute takes precedence. If the @value attribute is not specified and the <sort-as> element does not contain content, then the <sort-as> element has no effect.

Examples

Intro goes here ...

Comment by Kristen J Eberlein on 16 June 2019

I have not attempted to mak changes in this section with bold and line-through. The examples have not changed since DITA 1.3, although the markup for this example section has been modified.

Figure 8: Sorting glossary entries

The following code samples show how a glossary entry for the Chinese ideographic character for "big" might specify an effective sort phrase of "dada" (the Pin-Yin transliteration for Mandarin):

The <sort-as> element can be located within <glossterm>:

```
<glossentry id="gloss-dada">
  <glossterm><sort-as value="dada"/>&#x5927;&#x5927;</glossterm>
    <glossdef>Literally "big big".</glossdef>
</glossentry>
```

Or the <sort-as> element can be located within <prolog>:

```
<glossentry id="gloss-dada">
  <glossterm>&#x5927;&#x5927;</glossterm>
  <prolog>
    <sort-as>dada</sort-as>
  </prolog>
    <glossdef>Literally "big big".</glossdef>
</glossentry>
```

Figure 9: Sorting index entries

This following code sample shows an index entry for <data> that will be sorted as "data":

```
<indexterm>&lt;data&gt;<sort-as>data</sort-as></indexterm>
```