DITA Proposed Feature # 12020

Generic text element

Longer description

Expand the content model of base elements which can contain #PCDATA so that they can also contain a new text element.

Previous discussion of this proposed feature:

- Proposal for nesting keywords, from DITA 1.1

Statement of Requirement

Heavy users of conref have a need to create fragments of text which can be re-used in almost any context. This example is from the DITA 1.1 keyword nesting proposal:

> The keyword element is often used to store common text such as product or platform names. This is done because keywords are allowed in nearly all locations that allow text. However, it is not possible to combine common strings into one single keyword for reuse. For example, both “Product A” and “Platform B” are common strings, and are often used together as “Product A for Platform B” . To use the combined value, users must enter a new copy of each string in another keyword. Alternatively, they could always reference the two in text as `<keyword conref="#topic/product"/> for <keyword conref="#topic/platform"/>`. So, it is impossible to reuse the string “Product A for Platform B” without repeating text somewhere.
> http://www.oasis-open.org/committees/download.php/15522/IssueNumber02.html,

Similar issues happen with other elements such as term and specializations of these and of ph.

Specializers may have a requirement to disallow mixed content in an element, to ensure correct arity of child elements. For example, a specialization of keyword may need strict alternation of text with data elements:

```
<spec-keyword>
  text<spec-data-1/><spec-data-2>...</spec-data-2>
  text<spec-data-1/><spec-data-2>...</spec-data-2>
  text<spec-data-1/><spec-data-2>...</spec-data-2>
</spec-keyword>
```

This cannot be validated with XML Schema or DTD because mixed content models are too lax.

Use Cases

Conref of small pieces of text

Some elements such as keyword and term have a content model which does not allow further nesting of elements (discounting tm, which is usually not generic enough). Building such an element from multiple strings does not give the user a place to hang a conref.

Similarly, this problem can flow through to specializations, so that it is not possible to conref any text into a wintitle element.

With this proposal, text is available in all elements (including text itself) that don't contain ph. Strings can be built from pieces and inserted by conref into any context:

```
<topic id="strings">
  ...
  <body>
    <text id="productA">Product A</text>
    <text id="platformB">Platform B</text>
    <text id="AforB"><text conref="#strings/productA"/> for <text
conref="#strings/platformB"/></text>
  </body>
```

```
</topic>

<topic>
  <title>Using <keyword><text
conref="strings.xml#strings/AforB"/></keyword></title>
  ...
</topic>
```

Removing mixed content from a specialization

Mixed content models in DTD and XML Schema cannot prevent the appearance of text in undesired places:

```
<!ELEMENT spec-keyword ((#PCDATA | spec-data-1 | spec-data-2)*)>
```

By placing the text content inside a container element:

```
<spec-keyword>
  <text>text</text><spec-data-1/><spec-data-2>...</spec-data-2>
  <text>text</text><spec-data-1/><spec-data-2>...</spec-data-2>
  <text>text</text><spec-data-1/><spec-data-2>...</spec-data-2>
</spec-keyword>
```

validation can now be more strict:

```
<!ELEMENT spec-keyword ((text, spec-data-1, spec-data-2)*)>
```

Note: It may be desirable for text to be specializable so that it has a more appropriate name in the context of the specialization.

Scope

Minor. Content models for a few dozen elements require an additional entry.

Technical Requirements

The text element should be added to any element which allows #PCDATA but does not allow ph. In DITA 1.1, these elements are

- keyword
- term
- tm
- alt*
- indexterm*
- index-base*
- linktext*
- navtitle*
- searchtitle*
- author*
- source*
- publisher*
- copyrholder*
- category*
- prodname*
- brand*
- series*
- platform*
- prognum*
- featnum*
- component*

Items marked with an asterisk can contain keyword but not ph.

The text element should also be added to the content model of <ph> so that text can be conreffed into a phrase without additional semantics.

Specializations of these elements should also decide whether to include text. In DITA 1.1, these elements are

Bookmap
revisionid, year , month , day, edition, isbn, volume, person, organization, summary, printlocation, bookpartno, booknumber
Indexing domain
index-see, index-see-also, index-sort-as
Programming domain
option, parmname, synph, apiname, kwd, var, oper, delim, sep, repsep
Software domain
msgph, msgblock, filepath, userinput, systemoutput, msgnum, cmdname, varname
UI domain
wintitle, shortcut, uicontrol
Utilities domain
coords, shape
xNAL domain
honorific, firstname, middlename, lastname, generationidentifier, postalcode, country, contactnumber, otherinfo, addressdetails, locality, localityname, administrativearea, thoroughfare, emailaddress, url

The content model of text is

```
<!ELEMENT text ((#PCDATA | text)*)>
```

text contains all universal DITA attributes.

New or Changed Specification Language

This element requires no addition to the architectural specification.

The language specification should have an entry in the element reference for text. Here is a suggested description:

> The text element associates no semantics with its content. It exists to serve as a container for text where a container is needed (e.g., for conref, or for restricted content models in specializations). Unlike ph, text cannot contain images. The text element contains only text data, or nested text elements. All universal attributes are available on text.
>
> For contexts where ph is available, authors should use that element. Where ph is not available, text can be used to pull content by conref.

Costs

DTDs and Schemas must be updated.

Implementations will need to include processing for text. This may require implementations to handle XML fragments where they once needed to handle only strings. Fallback behaviour (flattening the XML a la xsl:value-of) is not appropriate for text if attributes like dir or translate or filtering properties are present.

Benefits

Greater re-use of boilerplate text by users and fewer (to users) arbitrary limitations. Specializers can have more control over content models by avoiding mixed content.