Hypertext Link Support Requirements for XSL

Christopher R. Maden
Senior Tools Specialist
O'Reilly & Associates, Inc.
90 Sherman Street
Cambridge, MA USA 02140
crism@oreilly.com

World-Wide Web Consortium XSL Working Group

Revision 0.1	11 May 1998	Initial draft.
Revision 0.2	12 May 1998	Filled out transclusion section.

Abstract: The XLink and XPointer specifications provide a level of hypertext functionality that has not previously been available for widespread use. To succeed as a stylesheet language under these new conditions, XSL must address certain styling problems introduced by rich hypertext, including stylistic treatment of link ends and styling of transcluded text.

Introduction

The XLink[1] and XPointer[2] specifications provide complex hypertext capabilities for XML documents. Two features with direct impact on the styling of XML documents are the creation of links whose link ends are in a different location from the link specification and the inclusion of one document, or a part thereof, in another as an apparent part of another document.

As part of its mission to offer functionality beyond that available in CSS, XSL should treat hypertext artifacts as first-class objects. Any link end known to the processor should be addressable, and the stylesheet should have access to any available information about that link end. XSL must also address the stylistic and intellectual property issues raised by the possibility of transclusion, as outlined in *Problems with Dynamically Assembled Document Portions, and Some Solutions*.[3]

Link End Formatting

In XLink, simple (or inline) links are represented by an element at their origin. Since XSL provides means for addressing elements in the document already, this adds no new requirements. However, extended links can create link ends that are at a different point in a document, or even in an altogether different document, from the point where the link is specified. A link end is not bound by element boundaries; it can cross them and it can start or end in the middle of an element. XSL must provide a means of treating link ends as units of information to be formatted, and must specify behavior when the same information participates in multiple links.

XLink also specifies a means of providing information about a link and its link ends: a role, a title, and behavioral qualities. XSL needs to specify the means of access to this information when formatting a link end; for example, to provide different formatting for link ends with different roles. XLink allows the author to rename key attributes to avoid name conflicts; XSL should provide unambiguous access to the semantics of the link, regardless of the names used to specify them.

The requirements for link end formatting, then, are:

- Location of link ends

  This could be accomplished through a reserved-name pseudo-element, such as `#linkend`, or through a new pattern, such as `<target-linkend>`.

- Identification of link end qualities

  This could happen on the pattern or construction sides, or both. Given an element-like rule for locating link ends, it seems logical to use a syntax similar to that used for attributes to get information about the link end pseudo-elements. The qualities specified by XLink are:

- role

  An unenumerated value identifying the nature of the link end or the link. The roles of both should be accessible.

- title

  An unenumerated value giving a possible caption for the link end. The titles of other link ends may be of more interest than the title of the current one; see below.

- show, actuate

  Primitive behavioral guides provided by XLink, to be used in the absence of other behavioral specifications (such as a stylesheet). These qualities should be accessible to the stylesheet; one test of sufficient power is the ability to implement a single rule that will provide the author's intended behavior for any link end.

- behavior

  An unenumerated property for "detailed instructions for traversal behavior." XSL expressions would not be unreasonable to place in the behavior attribute, and a stylesheet should be able to evaluate them.

- Other attributes

  The stylesheet should have access to any other attributes specified on the locator element. An important note is that those attributes may have names identical to those of XLink attributes, as the author can rename the XLink attributes; this implies a different syntax for referring to XLink link end qualities and to other locator attributes.

  Note that some characteristics of a link end, if specified on the link but not the link end, will inherit. An XSL stylesheet should not be able to distinguish between the two circumstances.

- Specification of multiple link end cascading

  Cascading and priority rules need to be defined for the event that a single datum participates in more than one link end. Different properties can be merged (*e.g.*, one link end specifies red, another underlined), but conflicting values for the same property can not be (if one link end specifies red and another blue).

- Output specifications sufficiently powerful for rich hypertext

  A link end from a single link may well have many possible destinations. XSL should provide flow objects capable of presenting complex hypertext reasonably to a user, such as a pop-up list of titled destinations, or even a set of such lists, grouped under the title of the link for which each item is a link end. The stylesheet may even wish to select a single destination out of a set, to force a user to cycle through the participants in a link in some mandated order.

- Default prototypes

  The XSL specification should provide an example, in XSL syntax, of rules that implement the link end's requested behavior. These samples may also provide some standard formatting, such as blue and underlined. Whether these examples should be only instructive, or whether they should also form some default rule in the absence of any other instruction, is open for debate. (I lean to the former.)

- Other link ends

  The rule for one link end may wish to have information about other ends of the same link: titles, roles, and maybe even URIs. This warrants further investigation, but should probably wait pending stability of XLink and a preliminary solution for other requirements.

Transclusion

The other very interesting feature of XLink is the ability to include another document, or a portion of another document, dynamically at a point in the present document. This was dubbed “transclusion” by hypertext pioneer Ted Nelson. Steve DeRose and I analyzed the stylistic problems presented by transclusion in the paper referenced earlier.[3] The conclusion that we reached is that transclusion problems can be decomposed into three classes of problems: proper display, addressing of transcluded data, and modification of the transcluded data.

In the area of display problems, we found that stylesheet control is necessary for proper handling of transclusion. First of all, transclusion may be desired in the absence of an explicit XLink instruction; for example, fetching the title of a referenced chapter is a form of transclusion. But alternately, the designer may legitimately wish to include the entire chapter within her document; the stylesheet must be able to make that distinction.

The stylesheet must also be able to exercise control over inheritance. For example, a document whose default stylesheet specifies twelve-point Caslon may transclude a poem whose default stylesheet specifies eleven-point Futura. The designer wishes to preserve the Caslon context; and further, to establish a ten-point and wider-margin context for the translucded content; but to permit further formatting from the poem's stylesheet, such as italics, to take place unhindered. The stylesheet needs to enable these specifications, and in general, to specify what other formatting is to be accepted from the transcluded text's stylesheet. Increased margins in the poetry should be honored, but not necessarily with the same values as given. These specifications will be complex, but must be feasible.

There's also a question of altered interpretation of the data itself. A reference to an item in a list may wish to use the item's original context, perhaps to generate the text “item 5: weld the sprocket to the widget.” But a series of items may be included in a list, and their numbering should be done in the context in which they appear instead.

A final question is that of modification. Our paper suggested that modification specifications might be contained within the element specifying the transclusion itself, but it's open to question whether acting on such specifications is the role of the linking engine or of the stylesheet. The stylesheet may wish to specify such modifications itself, and its processor is more likely to have implemented an expression language, making it a more likely candidate.

The transclusion requirements, then, are:

- Ability to request transclusion

  This is handled in part by the requirement above to be able to generically implement XLink behavioral specifications, but the stylesheet should be able to further modify the specified XPointer, such as to request the child of the target called `title`.

- Specification of context for translucded data

  This is not particularly difficult; given the ability to request transclusion, the stylesheet can create the equivalent of a DSSSL sequence or area with inheritable characteristics, and then place the transclusion with it.

- Specification of style selection rules for the transcluded data

  This is one of the most difficult requirements. If the transcluded data is from the same or a similar document type, the designer may wish to ignore the data's own stylesheet complete, or he may want to give it complete control; but more likely, he will want to specify that certain kinds of formatting are permissible (posture, weight, and maybe relative size changes) but not others (font family changes, page breaks); and he may wish to alter the interpretation of some changes (scaling absolute font size requests or margin changes, substituting for requested font families).

- Specification of apparent data context

  This could be as simple as a Boolean switch to specify that the transcluded data should be considered as though it had been referenced as an entity, though a more complex setting could be desirable (as for heterogeneous lists with items from different document types). This would probably bear further study.

- Modification of transcluded data

  This needs some debate. It's certainly part of a high-end stylesheet language to perform some manipulation of textual data, and modification of transcluded data may come “for free” from that capability. However, we may find that something more powerful is required, and have to decide whether this functionality belongs in a stylesheet language.

Notes

1. `<URL:http://www.w3.org/TR/1998/WD-xlink-19980303.html>`

2. `<URL:http://www.w3.org/TR/1998/WD-xptr-19980303.html>`

3. DeRose and Maden, in *Proceedings of SGML/XML '97*; also at `<URL:http://www.oreilly.com/people/staff/crism/transclu.html>`.