WD-xlink-19980303

XML Linking Language (XLink)

World Wide Web Consortium Working Draft 3-March-1998

This version:
http://www.w3.org/TR/1998/WD-xlink-19980303
Previous version:
http://www.w3.org/TR/WD-xml-link-970731
Latest version:
http://www.w3.org/TR/WD-xlink
Editors:
Eve Maler (ArborText) <elm@arbortext.com>
Steve DeRose (Inso Corp. and Brown University) <sderose@eps.inso.com>

Status of this document

This is a W3C Working Draft for review by W3C members and other interested parties. It is a draft document and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". A list of current W3C working drafts can be found at http://www.w3.org/TR.

This work is part of the W3C XML Activity (for current status, see http://www.w3.org/MarkUp/XML/Activity). For information about the XPointer language which is expected to be used with XLink, see http://www.w3.org/TR/WD-xptr.

See http://www.w3.org/TR/NOTE-xlink-principles for additional background on the design principles informing XLink.

Abstract

This document specifies constructs that may be inserted into XML resources to describe links between objects. It uses XML syntax to create structures that can describe the simple unidirectional hyperlinks of today's HTML as well as more sophisticated multi-ended and typed links.

XML Linking Language (XLink)

Version 1.0

Table of Contents

1. Introduction
   1.1 Origin and Goals
   1.2 Relationship to Existing Standards
   1.3 Terminology
   1.4 Notation
2. Locator Syntax
3. Link Recognition
4. Linking Elements
   4.1 Information Associated with Links
      4.1.1 Locators
      4.1.2 Link Semantics
      4.1.3 Remote Resource Semantics
      4.1.4 Local Resource Semantics
   4.2 Simple Links
   4.3 Extended Links
5. Extended Link Groups
6. Link Behavior
   6.1 The "Show" Axis
   6.2 The "Actuate" Axis
   6.3 Combinations of the "Show" and "Actuate" Axes
7. Attribute Remapping
8. Conformance

Appendices

A. Unfinished Work
   A.1 Structured Titles
B. References

---

1. Introduction

This document specifies constructs that may be inserted into XML resources to describe links between objects. A link, as the term is used here, is an explicit relationship between two or more data objects or portions of data objects. This specification is concerned with the syntax used to assert link existence and describe link characteristics. Implicit (unasserted) relationships, for example that of one word to the next or that of a word in a text to its entry in an on-line dictionary are obviously important, but outside its scope.

Links are asserted by elements contained in XML documents. The simplest case is very like an HTML A link, and has these characteristics:

- The link is expressed at one of its ends (similar to the A element in some document)
- Users can only initiate travel from that end to the other
- The link's effect on windows, frames, go-back lists, stylesheets in use, and so on is mainly determined by browsers, not by the link itself. For example, traversal of A links normally replaces the current view, perhaps with a user option to open a new window.
- The link goes to only one destination (although a server may have great freedom in finding or dynamically creating that destination).

While this set of characteristics is already very powerful and obviously has proven itself highly useful and effective, each of these assumptions also limits the range of hypertext functionality. The linking model defined here provides ways to create links that go beyond each of these specific characteristics, thus providing features previously available mostly in dedicated hypermedia systems.

1.1 Origin and Goals

Following is a summary of the design principles governing XLink:

1. XLink shall be straightforwardly usable over the Internet.
2. XLink shall be usable by a wide variety of link usage domains and of classes of linking application software.
3. The XLink expression language shall be XML.
4. The XLink design shall be prepared quickly.
5. The XLink design shall be formal and concise.
6. XLinks shall be human-readable.
7. XLinks may reside outside the documents in which the participating resources reside.
8. XLink shall represent the abstract structure and significance of links.
9. XLink must be feasible to implement.

1.2 Relationship to Existing Standards

Three standards have been especially influential:

- *HTML:* Defines several SGML element types that represent links.
- *HyTime:* Defines inline and out-of-line link structures and some semantic features, including traversal control and presentation of objects.
- *Text Encoding Initiative Guidelines (TEI P3):* Provide structures for creating links, aggregate objects, and link collections.

Many other linking systems have also informed this design, especially Dexter, FRESS, MicroCosm, and InterMedia.

1.3 Terminology

The following basic terms apply in this document.

element tree
: A representation of the relevant structure specified by the tags and attributes in an XML document, based on "groves" as defined in the ISO DSSSL standard.

inline link
: Abstractly, a link which serves as one of its own resources. Concretely, a link where the content of the linking element serves as a participating resource. HTML `A`, HyTime `clink`, and TEI `XREF` are all examples of inline links.

link
: An explicit relationship between two or more data objects or portions of data objects.

linking element
: An element that asserts the existence and describes the characteristics of a link.

local resource

The content of an inline linking element. Note that the content of the linking element could be explicitly pointed to by means of a regular locator in the same linking element, in which case the resource is considered remote, not local.

locator
Data, provided as part of a link, which identifies a resource.

multidirectional link
A link whose traversal can be initiated from more than one of its participating resources. Note that being able to "go back" after following a one-directional link does not make the link multidirectional.

out-of-line link
A link whose content does not serve as one of the link's participating resources . Such links presuppose a notion like extended link groups, which indicate to application software where to look for links. Out-of-line links are generally required for supporting multidirectional traversal and for allowing read-only resources to have outgoing links.

participating resource
A resource that belongs to a link. All resources are potential contributors to a link; participating resources are the actual contributors to a particular link.

remote resource
Any participating resource of a link that is pointed to with a locator.

resource
In the abstract sense, an addressable service or unit of information that participates in a link. Examples include files, images, documents, programs, and query results. Concretely, anything reachable by the use of a locator in some linking element. Note that this term and its definition are taken from the basic specifications governing the World Wide Web.

sub-resource
A portion of a resource, pointed to as the precise destination of a link. As one example, a link might specify that an entire document be retrieved and displayed, but that some specific part(s) of it is the specific linked data, to be treated in an application-appropriate manner such as indication by highlighting, scrolling, etc.

traversal
The action of using a link; that is, of accessing a resource. Traversal may be initiated by a user action (for example, clicking on the displayed content of a linking element) or occur under program control.

1.4 Notation

The formal grammar for locators is given using a simple Extended Backus-Naur Form (EBNF) location, as described in the XML specification.

2. Locator Syntax

The locator for a resource is typically provided by means of a Uniform Resource Identifier, or URI. XPointers can be used in conjunction with the URI structure, as fragment identifiers or queries, to specify a more precise sub-resource. XPointers can be used in conjunction with URIs to specify a more precise sub-resource.

A locator generally contains a URI, as described in IETF RFCs [IETF RFC 1738] and [IETF RFC 1808]. As these RFCs state, the URI may include a trailing *query* (marked by a leading "?"), and be followed by a "#" and a *fragment identifier*, with the query interpreted by the

host providing the indicated resource, and the interpretation of the fragment identifier dependent on the data type of the indicated resource.

In order to locate XML documents and portions of documents, a locator value may contain either a URI or a fragment identifier, or both. Any fragment identifier for pointing into XML must be an XPointer.

Special syntax may be used to request the use of particular processing models in accessing the locator's resource. This is designed to reflect the realities of network operation, where it may or may not be desirable to exercise fine control over the distribution of work between local and remote processors.

Locator

```
[1]   Locator ::= URI
                | Connector ( XPointer | Name)
                | URI Connector (XPointer | Name)
[2] Connector ::= '#' | '|'
[3]       URI ::= URIchar*
```

In this discussion, the term **designated resource** refers to the resource which an entire locator serves to locate. The following rules apply:

- The URI, if provided, locates a resource called the **containing resource** .
- If the URI is not provided, the containing resource is considered to be the document in which the linking element is contained.
- If an XPointer is provided, the designated resource is a **sub-resource** of the containing resource; otherwise the designated resource is the containing resource.
- If the `Connector` is followed directly by a `Name`, the `Name` is shorthand for the XPointer "`id(Name)`"; that is, the sub-resource is the element in the containing resource that has an XML ID attribute whose value matches the `Name`. This shorthand is to encourage use of the robust `id` addressing mode.
- If the connector is "#", this signals an intent that the containing resource is to be fetched as a whole from the host that provides it, and that the XPointer processing to extract the sub-resource is to be performed on the client, that is to say on the same system where the linking element is recognized and processed.
- If the connector is "|", no intent is signaled as to what processing model is to be used for accessing the designated resource.

Note that by definition, a URI includes an optional query component.

In the case where the URI contains a query (to be interpreted by the server), information providers and authors of server software are urged to use queries as follows:

Query

```
[4] Query ::= 'XML-XPTR=' ( XPointer | Name)
```

3. Link Recognition

The existence of a link is asserted by a linking element. Linking elements must be recognized reliably by application software in order to provide appropriate display and behavior. There are several ways link recognition could be accomplished: for example, reserving element type names, reserving attributes, or leaving the matter of recognition entirely up to stylesheets and application software. Reserving attributes provides a balance between giving users control of their own markup language design and keeping the important structural fact "is a link" explicit within documents. Therefore, XLink linking-related elements are recognized based on the use of a designated attribute named `xml:link`. Possible values are `simple` and `extended` (which identify linking elements), as well as `locator`, `group`, and `document` (which identify other related types of elements). An element in whose start-tag such an attribute appears is to be treated as an element of the indicated XLink type as dictated by this specification. For example:

```
<A xml:link="simple" href="http://www.w3.org/">The W3C</A>
```

Note: Subject to definitions to be developed in related standards, the methods described in "7. Attribute Remapping" may be used to rename the reserved attribute.

There are two mechanisms that may be used to associate the `xml:link` and `xml:attributes` attributes with a linking element. The simplest is to provide the attribute explicitly in a start-tag. A less verbose method is to use XML's facilities for declaring default attribute values. For example, the following attribute-list declaration would indicate that all instances of the `A` element in the current document are XLink simple links:

```
<!ATTLIST A xml:link CDATA #FIXED "simple">
```

4. Linking Elements

XLink defines two types of linking element:

- A simple link, which is usually inline and always one-directional
- A much more general extended link, which may be either inline or out-of-line and must be used for multidirectional links, links originating from read-only resources, and so on.

Both kinds of links can have various types of information associated with them.

4.1 Information Associated with Links

The following information can be associated with a link and its resources:

- One or more locators to identify the remote resources participating in the link; a locator is required for each remote resource
- Semantics of the link
- Semantics of the remote resources
- Semantics of the local resource , if the link is inline

This information is supplied in the form of attributes on linking elements. In the following sections, parameter entities are used to group these attributes.

4.1.1 Locators

A locator string identifies a participating resource. A link must supply a locator for each remote resource.

A locator takes the form of an attribute called `href`. Following is a sample declaration of this attribute, enclosed in a `locator.att` parameter entity.

```
<!ENTITY % locator.att
  "href          CDATA              #REQUIRED"
>
```

4.1.2 Link Semantics

The following semantic information can be provided for a link:

- Whether the link is inline

  If the link is inline, its content counts as a local resource of the link. (However, any locator subelements inside the linking element are *not* considered part of the local resource; they are simply part of the linking element machinery.) If the link is out-of-line, its content does not count as a local resource. Every link is either inline or out-of-line. The inline status of a link is indicated with an attribute called `inline`. It can have the value `true` (the default) or `false`.
- The **role** of the link, to identify to application software the meaning of the link

  Links express various kinds of conceptual relationships between the data objects or portions they connect, in terms of significance to the author and user. Some links may be criticisms, others add support or background, while still others might provide access to demographic information about a data object (its author's name, version number, etc), or to navigational tools such as index, glossary, and summary. To indicate the part that a link plays in representing information, a link author can optionally provide a string identifying the link's role. The role is indicated with an attribute called `role`. (Note that each resource participating in a link may also be given its own role, as described in "4.1.3 Remote Resource Semantics".)

Following are sample declarations of these attributes, enclosed in a `link-semantics.att` parameter entity.

```
<!ENTITY % link-semantics.att
  "inline        (true|false)       'true'
   role          CDATA              #IMPLIED"
>
```

Because simple links have an attribute called `role` that has a different function, they cannot have a `role` attribute for link semantics. Following is a `simple-link-semantics.att`

parameter entity declaration for use in simple linking elements.

```
<!ENTITY % simple-link-semantics.att
  "inline        (true|false)        'true'"
>
```

4.1.3 Remote Resource Semantics

The following semantic information can be provided for the remote resources of a link:

- The role of the resource, to identify to application software the part it plays in the link

  (Note that a link as a whole may also be given its own role, as described in "4.1.2 Link Semantics".) A link author can optionally provide role information in an attribute called `role`.
- A title for the resource, to serve as a displayable caption that explains to users the part the resource plays in the link

  A link author can optionally provide title information in an attribute called `title`. XLink does not require that application software make any particular use of title information.
- Behavior policies to use in traversing to this resource

  A link author can optionally use attributes called `show` and `actuate` to communicate general policies concerning the traversal behavior of the link. The `show` attribute can have one of the values `new`, `replace`, and `embed`; the `actuate` attribute can have one of the values `auto` and `user`. A link author can also optionally use an attribute called `behavior` to communicate detailed instructions for traversal behavior. The contents, format, and meaning of this attribute are unconstrained. (See "6. Link Behavior" for more information on the behavior-related attributes.)

Following are sample declarations of these attributes, enclosed in a `remote-resource-semantics.att` parameter entity.

```
<!ENTITY % remote-resource-semantics.att
  "role          CDATA               #IMPLIED
   title         CDATA               #IMPLIED
   show          (embed|replace|new) #IMPLIED
   actuate       (auto|user)         #IMPLIED
   behavior      CDATA               #IMPLIED"
>
```

4.1.4 Local Resource Semantics

The following semantic information can be provided for the local resource of a link, if the link is inline:

- The role of the resource, to identify to application software the part it plays in the link

  (Note that a link as a whole may also be given its own role, as described in "4.1.2 Link Semantics".) A link author can optionally provide role information in an attribute

called `content-role`.
- A title for the resource, to serve as a displayable caption that explains to users the part the resource plays in the link

  A link author can optionally provide title information in an attribute called `content-title`. XLink does not require that application software make any particular use of title information.

Following are sample declarations of these attributes, enclosed in a `local-resource-semantics.att` parameter entity.

```
<!ENTITY % local-resource-semantics.att
  "content-role  CDATA               #IMPLIED
   content-title CDATA               #IMPLIED"
>
```

4.2 Simple Links

Simple links can be used for purposes that approximate the functionality of a basic HTML `A` link, but they can also support a limited amount of additional functionality. Simple links have only one locator and thus, for convenience, combine the functions of a linking element and a locator into a single element. As a result of this combination, the simple linking element offers both a locator attribute and all the link and resource semantic attributes.

Following is a sample declaration for a simple link, showing all the possible XLink-related attributes it may have (using the parameter entities provided in "4.1 Information Associated with Links"). The `xml:link` attribute value for a simple link must be `simple`.

```
<!ELEMENT simple ANY>
<!ATTLIST simple
    xml:link      CDATA               #FIXED "simple"
    %locator.att;
    %remote-resource-semantics.att;
    %local-resource-semantics.att;
    %simple-link-semantics.att;
>
```

There are no constraints on the contents of a simple linking element. In the sample declaration above, it is given a content model of `ANY` to indicate that any content model or declared content is acceptable. In a valid document, every element that is significant to XLink must still conform to the constraints expressed in its governing DTD.

Following is an example of a simple link:

```
<mylink xml:link="simple" title="Citation"
   href="http://www.xyz.com/xml/foo.xml" show="new"
   content-role="Reference">as discussed in Smith(1997)</mylink>
```

This example `mylink` element might have the following element and attribute-list declarations:

```
<!ELEMENT mylink (#PCDATA)>
<!ATTLIST mylink
    xml:link      CDATA               #FIXED "simple"
    href          CDATA               #REQUIRED
    content-role  CDATA               #IMPLIED
>
```

Note that it is meaningful to have an out-of-line simple link, although such links are uncommon. They are called "one-ended" and are typically used to associate discrete semantic properties with locations. The properties might be expressed by attributes on the link, the link's element type name, or in some other way, and are not considered full-fledged resources of the link. Most out-of-line links are extended links, as these have a far wider range of uses.

4.3 Extended Links

An **extended link** differs from a simple link in that it can connect any number of resources, not just one local resource (optionally) and one remote resource, and in that extended links are more often out-of-line than simple links.

The additional capabilities of extended links are required for:

- Enabling outgoing links in documents that cannot be modified to add an inline link
- Creating links to and from resources in formats with no native support for embedded links (such as most multimedia formats)
- Applying and filtering sets of relevant links on demand
- Enabling other advanced hypermedia capabilities

Application software might provide traversal among all of a link's participating resources (subject to semantic constraints outside the scope of this specification) and might signal the fact that a given resource or sub-resource participates in one or more links when it is displayed (even though there is no markup at exactly that point to signal it).

A linking element for an extended link contains a series of child elements that serve as locators. Because an extended link can have more than one remote resource, it separates out linking itself from the mechanisms used to locate each resource (whereas a simple link combines the two).

The linking element itself retains those attributes relevant to the link as a whole and to its local resource, if any. Following is a sample declaration for an extended link (using the parameter entities provided in "4.1 Information Associated with Links"). The `xml:link` attribute value for an extended link must be `extended`.

```
<!ELEMENT extended ANY>
<!ATTLIST extended
    xml:link      CDATA               #FIXED "extended"
    %link-semantics.att;
    %local-resource-semantics.att;
>
```

Attributes relevant to remote resources are expressed on the corresponding contained locator elements. Each remote resource can have its own semantics in relation to the link as a whole. Following is a sample declaration for a locator element, showing all the possible XLink-related attributes it may have (using the parameter entities provided in "4.1 Information Associated with Links"). The `xml:link` attribute value for a locator element must be `locator`.

```
<!ELEMENT locator ANY>
<!ATTLIST locator
    xml:link      CDATA               #FIXED "locator"
    %locator.att;
    %remote-resource-semantics.att;
>
```

Following is an example of an out-of-line extended link:

```
<commentary xml:link="extended" inline="false">
   <locator href="smith2.1" role="Essay"/>
   <locator href="jones1.4" role="Rebuttal"/>
   <locator href="robin3.2" role="Comparison"/>
</commentary>
```

For convenience, defaults for the semantic attributes on locator elements can be specified on the linking element that contains them. If any such attribute is omitted from a locator element, the value provided on the containing linking element is to be used. Following is a sample declaration for an extended link (using the parameter entities provided in "4.1 Information Associated with Links") showing all the possible XLink-related attributes it may have, including the remote resource semantic attributes.

```
<!ELEMENT extended ANY>
<!ATTLIST extended
    xml:link      CDATA               #FIXED "extended"
    %link-semantics.att;
    %local-resource-semantics.att;
    %remote-resource-semantics.att;
>
```

The content of a linking element typically consists only of locator elements; however, the declaration as `ANY` indicates that any other content may be added. (In a valid document, every element that is significant to XLink must still conform to the constraints expressed in its governing DTD.) Only locator elements that are direct children of the linking element define resources linked by that linking element.

A key issue with out-of-line extended links is how linking application software can manage and find them, particularly when they are stored in completely separate documents from those in which their participating resources appear. XLink provides a mechanism for identifying relevant link-containing documents, which is discussed in "5. Extended Link Groups".

5. Extended Link Groups

Hyperlinked documents are often best processed in groups rather than one at a time. If it is

desired to highlight resources to advertise that traversal can be initiated, and if at the same time out-of-line links are being used, it may be an absolute requirement to read other documents to find these links and discover where the resources are.

In these cases, an **extended link group** element, a special kind of extended link, may be used to store a list of links to other documents that together constitute an interlinked group. Each such document is identified by means of an **extended link document** element, a special kind of locator element.

Following are sample declarations for extended link group and extended link document elements, showing all the possible XLink-related attributes they may have (using the parameter entities provided in "4.1 Information Associated with Links"). The `xml:link` attribute value for an extended link group element must be `group`, and the value for an extended link document element must be `document`.

```
<!ELEMENT group (document*)>
<!ATTLIST group
    xml:link      CDATA               #FIXED "group"
    steps         CDATA               #IMPLIED
>
<!ELEMENT document EMPTY>
<!ATTLIST document
    xml:link      CDATA               #FIXED "document"
    %locator.att;
>
```

The `steps` attribute may be used by an author to help deal with the situation where an extended link group directs application software to locate another document, which proves to contain an extended link group of its own. There is a potential for infinite regress, and yet there are situations where processing several levels of extended link groups is useful. The `steps` attribute should have a numeric value that serves as a hint from the author to any link processor as to how many steps of extended link group processing should be undertaken. It does not have any normative effect.

For example, should a group of documents be organized with a single "hub" document containing all the out-of-line links, it might make sense for each non-hub document to contain an extended link group containing only one reference to the hub document. In this case, the best value for `steps` would be `2`.

6. Link Behavior

Link formatting and link behavior are inextricably connected. In general, formatting involves the appearance or treatment of the link prior to any user action, such as choice of font, color, icons, and other devices to show that a link is present. Behavior focuses on what happens when the link is traversed, such as opening, closing, or scrolling windows or panes; displaying the data from various resources in various ways; testing, authenticating, or logging user and context information; or executing various programs.

XLink does not provide mechanisms for controlling link formatting because it is considered to fall into the domain of stylesheets. Link behavior should ideally also be determined by rules based on link types, resource roles, user circumstances, and other factors. However,

XLink does provide a few very general behavior mechanisms because they are commonly considered to reflect major or invariant semantics of link types.

The mechanism that XLink provides allows link authors to signal certain intentions as to the timing and effects of traversal. Such intentions can be expressed along two axes, labeled `show` and `actuate`. These are used to express *policies* rather than *mechanisms* ; any link-processing application software is free to devise its own mechanisms, best suited to the user environment and processing mode, to implement the requested policies.

In many cases, much finer control over the details of traversal behavior, of the type that existing hypertext software typically provides, will be desired. Such fine control of link behavior is outside the scope of this specification. However, the `behavior` attribute is provided as a standard place for authors to provide, and in which application software may look, for detailed behavioral instructions.

6.1 The "Show" Axis

The `show` attribute is used to express a policy as to the context in which a resource that is traversed to should be displayed or processed. It may take one of three values:

`embed`
: Indicates that upon traversal of the link, the designated resource should be embedded, for the purposes of display or processing, in the body of the resource and at the location where the traversal started.

`replace`
: Indicates that upon traversal of the link, the designated resource should, for the purposes of display or processing, replace the resource where the traversal started.

`new`
: Indicates that upon traversal of the link, the designated resource should be displayed or processed in a new context, not affecting that of the resource where the traversal started.

6.2 The "Actuate" Axis

The `actuate` attribute is used to express a policy as to when traversal of a link should occur. It may take one of two values:

`auto`
: Indicates that the resource in question should be retrieved when any of the other resources of the same link is encountered, and that the display or processing of the initiating resource is not considered complete until this is done. All auto resources are retrieved in the order specified.

`user`
: Indicates that the resource should not be presented until there is an explicit external request for traversal.

6.3 Combinations of the "Show" and "Actuate" Axes

Each combination of the `show` and `actuate` attributes is meaningful. Perhaps the least obvious is `show="replace"` combined with `actuate="auto"`; this could be used in "forwarding"

type applications, where when one anchor is display, the other(s) are to replace it without user intervention. Since XLink provides only the most general semantics for links, details of presentation, such as a time delay or beep before forwarding, can be specified on a per-application basis using a style language.

7. Attribute Remapping

XLink provides many attributes that can be attached to linking elements to describe various aspects of links, and each has a default name. It may be desired to use existing elements in XML documents as linking elements, but such elements might already have attributes whose names conflict with those described in this document. To avoid collisions, user-chosen attribute names can be mapped to the default names using the `xml:attributes` attribute.

This attribute must contain an even number of white-space-separated names, which are treated as pairs. In each pair, the first name must be one of the default XLink names (`role`, `href`, `title`, `show`, `inline`, `content-role`, `content-title` , `actuate`, `behavior`, `steps`). The second name, when recognized in the document, will be treated as though it were playing the role assigned to the first. For example, consider a DTD with the following declaration:

```
<!ELEMENT TEXT-BOOK ANY>
<!ATTLIST TEXT-BOOK
        title      CDATA                #IMPLIED
        role       (PRIMARY|SUPPORTING) #IMPLIED
>
```

If it were desired to use this as a simple link, it would be necessary to remap a couple of attributes. This could be accomplished in the internal subset:

```
<!ATTLIST TEXT-BOOK
        xml:link       CDATA            #FIXED "simple"
        xml:attributes CDATA
                       #FIXED "title xl-title role xl-role"
>
```

Then in the document, the following would be recognized as a simple link:

```
<TEXT-BOOK title="Compilers: Principles, Techniques, and Tools"
           role="PRIMARY" xl-title="Primary Textbook for the Course"
           xl-role="ONLINE-PURCHASE"
           href="/cgi/auth-search?q="+Aho+Sethi+Ullman"/>
```

8. Conformance

An element conforms to XLink if:

1. The element has an `xml:link` attribute whose value is one of the attribute values prescribed by this specification, and
2. the element and all of its attributes and content adhere to the syntactic requirements

imposed by the chosen `xml:link` attribute value, as prescribed in this specification.

Note that conformance is assessed at the level of individual elements, rather than whole XML documents, because XLink and non-XLink linking mechanisms may be used side by side in any one document.

An application conforms to XLink if it interprets XLink-conforming elements according to all required semantics prescribed by this specification and, for any optional semantics it chooses to support, supports them in the way prescribed.

Appendices

A. Unfinished Work

A.1 Structured Titles

The simple title mechanism described in this draft is insufficient to cope with internationalization or the use of multimedia in link titles. A future version will provide a mechanism for the use of structured link titles.

B. References

XPTR
Eve Maler and Steve DeRose, editors. XML Pointer Language (XPointer) V1.0. ArborText, Inso, and Brown University. Burlington, Seekonk, et al.: World Wide Web Consortium, 1998. (See http://www.w3.org/TR/WD-xptr .)

ISO/IEC 10744
ISO (International Organization for Standardization). ISO/IEC 10744-1992 (E). Information technology --Hypermedia/Time-based Structuring Language (HyTime). [Geneva]: International Organization for Standardization, 1992. Extended Facilities Annex. [Geneva]: International Organization for Standardization, 1996. (See http://www.ornl.gov/sgml/wg8/hytime/html/is10744r.html).

IETF RFC 1738
IETF (Internet Engineering Task Force). RFC 1738: Uniform Resource Locators. 1991. (See http://www.w3.org/Addressing/rfc1738.txt).

IETF RFC 1808
IETF (Internet Engineering Task Force). RFC 1808: Relative Uniform Resource Locators. 1995. (See http://www.w3.org/Addressing/rfc1808.txt).

TEI
C. M. Sperberg-McQueen and Lou Burnard, editors. Guidelines for Electronic Text Encoding and Interchange. Association for Computers and the Humanities (ACH), Association for Computational Linguistics (ACL), and Association for Literary and Linguistic Computing (ALLC). Chicago, Oxford: Text Encoding Initiative, 1994.

CHUM
Steven J. DeRose and David G. Durand. 1995. "The TEI Hypertext Guidelines." In Computing and the Humanities 29(3). Reprinted in Text Encoding Initiative:

Background and Context, ed. Nancy Ide and Jean Véronis, ISBN 0-7923-3704-2.