Mitschrift der Vorträge:

- **XLink**
- **User Agent Accessibility Guidelines**
- **XML Events**

VL - Netzbasierte Informationssysteme

Thomas Sliwa

Freie Universität Berlin
Fachbereich Mathematik und Informatik
Institut für Informatik

Takustr. 9, 14195 Berlin
E-Mail: sliwa@mi.fu-berlin.de
www: http://page.mi.fu-berlin.de/∼sliwa

22. Dezember 2004

Inhaltsverzeichnis

1 XLink

Vorgetragen am 25.11.2004 von Daria Schymura und Jan Felix Breuer.

Der Vortrag unterteilte sich in folgende Teile:

- Dem Vorstellen der Konzepte und des Designs von XLink
- Ein Beispiel
- Den Beschränkungen von XLink
- Der Verbreitung von XLink

1.1 Konzepte und Design (Beispiel integriert)

XLink ist eine W3C Recommendation aus dem Jahre 2000, um in XML und den daraus abgeleiteten Standards Verknüpfungen zu erstellen. Bei der Erstellung der Recommendation hielt man sich an schon bestehende Standards, wie HTML, HyTime und Text Encoding Initiative Guidelines. Ziel von XLink ist es einerseits einfach gerichtete Verknüpfungen, wie z.B. aus HTML bekannt, aber auch komplex strukturierte Verknüpfungen zwischen XML-Dokumenten und Elementen einheitlich zu definieren. Dabei stellt der XLink Namensraum eine Menge von Attributen bereit, mittels derer beliebige Elemente verknüpft werden können. Es gibt also in XLink keine neuen Elemente, sondern nur Attributdefinitionen.

XLink bietet neben dem einfachen Einfügen in XML-Dokumente weitere konzeptionelle Vorteile. So können Verknüpfungen Metadaten zugewiesen werden, wodurch man semantische Zusammenhänge zwischen Quell- und Zieldokument ausdrücken kann. Des Weiteren ist man in der Lage mehr als zwei Dokumente mit einer Verknüpfung zu verknüpfen und Quell- sowie Zieldokument müssen nicht an dem Ort der Verknüpfung liegen. Diese letzte Eigenschaft ermöglicht es so genannte Linkdatenbanken aufzubauen.

Wie im vorherigen Absatz erwähnt, können mit XLink mehr als zwei Dokumente mit einer Verknüpfung verknüpft werden. Beim Traversieren werden allerdings immer nur zwei Dokumente benutzt. Dies bedingt dann, dass man Informationen über die Art und Weise des Traversierens angibt. Genau diese Informationen werden in so genannten arcs angegeben. Somit beinhaltet das verknüpfte Element arcs für jedes Dokumentenpaar, das Teil der Verknüpfung sein soll. Man unterscheidet bei den arcs dann noch zwischen auswärts-, einwärtsgehenden und third-party, wobei bei auswärtsgehenden das Ziel durch eine URI, bei einwärtsgehenden die Quelle durch eine URI und bei third-party Quelle sowie Ziel durch URIs angegeben werden. Dokumente mit nur einwärtsgehenden und third-party arcs ermöglichen dann den Aufbau der vorher erwähnten Linkdatenbanken.

Um einerseits einfach zu beherrschen und andererseits flexibel in der Komplexität von Verknüpfungsstrukturen zu sein, gibt es bei XLink verschiedene Arten von Verknüpfungstypen. Diese werden mittels des Attributs type angegeben, welches folgende Werte annehmen kann:

- simple
- extended
- locator
- arc
- resource
- title
- none

Dabei ist zu beachten, dass nur die Typen extended und simple Verknüpfungen generieren und die anderen erwähnten Typen zur Definition von Verknüpfungen des Typs extended verwendet werden. Weitere Attributangaben, die zu einer Verknüpfung gehören, müssen dann dem Typ entsprechend angegeben werden.

Als weitere Attribute neben type, sieht die Recommendation die folgenden neun vor (in Klammern die Bedeutung):

- href (Lokalisierung von Dokumenten)
- role (semantische Bedeutung)
- arcrole (semantische Bedeutung)
- title (semantische Bedeutung)
- show (Verhaltenssteuerung beim Traversieren)
- actuate (Verhaltenssteuerung beim Traversieren)
- label (Spezifizierung der Traversierung)
- from (Spezifizierung der Traversierung)
- to (Spezifizierung der Traversierung)

Diese Attribute werden vor allem bei extended-Verknüpfungen benutzt, um die Verkettung mehrerer Dokumente zu erstellen und sinnvoll zu gliedern. Anhand des im Vortrag vorgestellten Beispiels wird dies für die Attribute mit semantischer Bedeutung einleuchtend. Mehrere Ressourcen (in diesem Fall eine Vorlesung, ein Student, ein Tutor und die Note des Studenten) werden in einem Element Teilnahme zusammengefasst. Dieses Element ist nun eine Verknüpfung des Typs extended. Die einzelnen Ressourcen werden dann Typisiert, d.h. es wird angegeben, ob sie extern oder intern vorliegen, und mit passenden Metadaten versehen. Schließlich werden noch die arcs erstellt mittels derer die konkreten Zusammenhänge ausgedrückt werden. Die Attribute zur Verhaltenssteuerung beim Traversieren bieten die Möglichkeit anzugeben wie das Zieldokument angezeigt werden soll und wann der arc traversiert werden soll.

1.2 Beschränkungen

Neben den Vorteilen hat XLink allerdings auch gewisse Beschränkungen. So ist es nicht möglich innerhalb einer Verknüpfung mehrere arcs zwischen zwei Dokumenten A und B zu haben. Dies ist auch nicht mit dem Attribut arcrole zu umgehen. Ergo haben zwei Dokumente, die mittels eines arcs verbunden sind, auch nur eine Rolle zueinander. Als zweite Einschränkung müssen arcroles immer eine absolute URI enthalten, da arcroles einem arc eine semantische Bedeutung zuordnen sollen und nicht z.B. irgendwelche Werte. Die letzte Beschränkung von Xlink ist im Zusammenhang mit XHTML gegeben. Benutzt man XLink in XHTML so ist man gezwungen das href-Attribut in beiden Namespaces mit einem Wert zu belegen, damit man beiden Namespaces genügt.

1.3 Verbreitung

Laut W3C gibt es sieben Produkte, die XLink implementieren. Darunter befinden sich die Browser Mozilla und Amaya, die allerdings nur das Traversieren von simple-Verknüpfungen unterstützen. Nur zwei der vorgestellten Implementierungen sind in der Lage mit extended-Verknüpfungen umzugehen. Davon darf allerdings eines nicht die Landesgrenzen von Japan überschreiten und die Entwicklung der zweiten ist anscheinend eingestellt, da sich keine Informationen darüber mehr im Internet befinden.

1.4 Offene Fragen

Es gab in der abschließenden Diskussion nur eine Frage: Muß sich hinter dem Attribut role eine tatsächliche Ressource befinden? Nein, die Angabe verhält sich analog zu einer Namensraumangabe. Es kann ein Dokument vohanden sein, muß aber nicht.

2 User Agent Accessibility Guidelines 1.0

Vorgetragen am 2.12.2004 von Alina Elias Zaleta und Eric Talbot.

Der Vortrag unterteilte sich in folgende Teile:

- Die Definition von UUAG 1.0
- Eine Einführung
- Der Darstellung der UUAG innerhalb des W3C
- Den UAAG 1.0 - Richtlinien

2.1 Definition

Die W3C Recommendation User Agent Accessibility Guidelines in der Version 1.0 ist Teil der Web Accessibility Initiative (WAI). Die Recommendation stellt Richtlinien für die Entwicklung von Web-Benutzerprogrammen auf. Die Unterstützung der UAAG stellt behinderten Menschen Benutzerprogramme bereit, mittels derer sie Zutritt zum Web haben.

2.2 Einführung

Um den Zugang zum Web auch für behinderte Menschen sicherzustellen, braucht man Richtlinien auf verschiedenen Ebenen. Neben den Benutzerprogrammen, müssen die Mensch-Computerschnittstelle sowie die Webinhalte selber auch behindertengerecht sein. So gibt es neben der UAAG spezielle Hardware, wie Brailledrucker, für die Mensch-Computerschnittstelle aber auch weitere W3C Recommendations für die barrierefreie Webprogrammierung. Nur wenn alle Ebenen implementiert sind, ist ein Webinhalt barrierefrei zu nutzen.

2.3 W3C

Innerhalb des W3C gibt es wie bereits erwähnt einen eigene Initiative für die Zugänglichkeit des Webs. Innerhalb dieser Web Accessibility Initiative (WAI) gibt es wiederum acht verschiedene Arbeitsgruppen, die sich mit den im vorherigen Abschnitt befassten Ebenen des Zugangs beschäftigen. Diese kümmern sich z.B. um Richtlinien für die Webinhaltentwicklung oder aber auch um die Verbreitung des Thema Zugang des Webs für behinderte Menschen.

2.4 Die Richtlinien

Die UAAG 1.0 stellt 12 Richtlinien auf, die Entwickler von Benutzerprogrammen befolgen sollen. Dabei hat jede Richtlinie so genannte Checkpoints, die priorisiert implementiert werden sollen. Im Folgenden werden die einzelnen Richtlinien vorgestellt ohne

aber detailliert auf die einzelnen Checkpoints einzugehen, da dies den Rahmen dieser Zusammenfassung sprengen würde.

2.4.1 Unabhängigkeit der Ein- und Ausgabevorrichtungen unterstützen

Es soll sichergestellt sein, dass das Benutzerprogramm die Ein- und Ausgabeschnittstelle des Benutzersystems unterstützt. Damit soll gewährleistet werden, dass der Benutzer mit dem Benutzerprogramm interagieren kann und somit mit dem Webinhalt. Dazu muß das Benutzerprogramm vollständig per Tastatur zu bedienen sein, Eventhandler unterstützen und Nachrichten als Text bereitstellen.

2.4.2 Benutzerzugang zum Webinhalt sicherstellen

Die Entwickler sollen sicherstellen, dass Benutzer Zugang zum gesamten Inhalt einer Website haben. Somit vor allem zu Teilinhalten, die speziell den Anforderungen der Recommendation Web Content Accessibility Guidelines 1.0 erstellt wurden und somit behindertengerecht sind. Erreicht werden soll dies unteranderem durch die Darstellung des jeweiligen Inhaltes gemäß seiner Spezifikation, Textansichten, der Ausgabe eingeschränkter Inhalte, der Ermöglichung einer zeitunabhängigen Darstellung, Beschreibungstexten von nicht textuellen Inhalten und der Berücksichtigung von Synchronisationssignalen, um nur die am höchsten priorisierten zu nennen.

2.4.3 Konfiguration ermöglichen, Webinhalte auszublenden, die den Zugang verringern

Der Benutzer soll die Möglichkeit haben Teilinhalte per Konfiguration auszublenden, wie z.B. Bild-, Audio-, Videoinhalte oder Scriptausgaben. Somit soll gewährleistet werden, dass es nicht zu Verwirrungen des Benutzers bzw. der Einschränkung des Zugangs zum Inhalt kommt.

2.4.4 Dem Benutzer die Kontrolle über die Ausgabemöglichkeit gewähren

Mittels dieser Richtlinie soll der Benutzer die Kontrolle über die Darstellungsform von Inhalten erlangen. Es soll ihm ermöglicht werden die Schriftform, -farbe und -grösse selbst zu bestimmen und diese auch gegenüber den Inhalt- und Benutzerprogramm eingestellten Werten durchzusetzen. Diese Kontrolle soll für Text- und Multimediainhalte zu Verfügung stehen.

2.4.5 Benutzerkontrolle über das Interface des Benutzerprogramms sicherstellen

Der Benutzer soll die Kontrolle über die Darstellung de Benutzerprogramms haben, d.h. er legt fest, wie Öffnen, Schließen und Fokuswechsel von Ansichten sowie Formulareingaben von dem Benutzerprogramm hantiert werden. So soll es beispielsweise keine automatischen Fokuswechsel bei Pop-Ups durch das Programm geben oder immer eine Nachfrage bei Formulareingaben geben.

2.4.6 Eine interoperabele Programmschnittstelle soll implementiert sein

Das Benutzerprogramm soll es per Programmschnittstelle ermöglichen mit anderen Komponenten, vor allem welchen, die für den Zugang zu Webinhalten sorgen, zu kommunizieren. So soll es Schnittstellen für den Zugriff auf Webinhalte sowie Funktionen desBenutzerprogrammss geben. Zusätzlich sollen schon vorhandene Programmschnittstellen und Standards, soweit implementiert, ebenfalls verfügbar sein. Hiermit sind dann z.B. Tastaturlayouts oder DOM gemeint.

2.4.7 Beachtung der Betriebsumgebung

Benutzerprogramme sollen die jeweilige Betriebsumgebung beachten, d.h. es sollenvorhandenee systemweite Einstellungen der Ein- und Ausgabegeräte, wie z.B. Standardtastaturabkürzungen, beachtet und übernommen werden. So soll eine einheitliche Bedienung eines Systems gewahrleistett werden.

2.4.8 Webzugangsfördernde Standards sollen implementiert werden

Schon vorhandene Standards, die den behindertengerechten Zugriff auf Webinhalte gewährleisten, sollen implementiert werden. Neben diesen zugangsförderneden Standards sollen aber auch weitere Webtechniken (HTML, XHTML, CSS, usw.) standardkonform implementiert werden, um die richtige Darstellung zu erreichen.

2.4.9 Navigationsmechanismen bereitstellen

Das Benutzerprogramm muß Navigationsmöglkichkeiten bereitstellen, die es auch behinderten Menschen ermöglicht sich in Webinhalten zu bewegen. So muß es dem Benutzer möglich sein mit Abkürzungen den Fokus auf einer Seite auf Verweise, Überschriften, Tabellen etc. zu setzen. Des Weiteren muß es eine History geben, um zwischen besuchten Inhalten zuwechselnn.Schließlich soll es eine Suchfunktion für den dargestellten Inhalt geben, um direkt auf Teilinhalte zu navigieren.

2.4.10 Orientierungshilfen bereitstellen

Das Benutzerprogramm soll dem Benutzer die Orientierung auf Webinhalten gewährleisten. Daher soll das Hervorheben von Inhalten, wie Verweisen bzw. schon besuchten Verweisen, oder das Berücksichtigen von Formatierungen, wie Überschriften oder Tabellen- und Bilbeschreibungen, implementiert sein. Außerdem soll über die Position innerhalb eines dargestellten Inhalts informiert werden.

2.4.11 Konfiguration und Anpassung sicherstellen

Der Benutzer soll das Programm nach persönlichen Interessen Konfigurieren und Anpassen können und diese Einstellungen speichern können. Es soll dem Benutzer so möglich sein, die Standardeinstellungen zu ändern sowie eigene Befehlseinstellungen festzulegen, die die Bedienung beeinflussen, ergo das Öffnen einer Webresource oder

die Steuerung multimedialer Inhalte. Auch das Interface soll frei konfigurierbar sein, in Bezug auf Aussehen und Positionierung.

2.4.12 Dokumentation und Hilfe verfügbar machen

Die Dokumentation und Hilfe des Benutzerprogramms müssen Benutzern zugänglich gemacht werden. Dabei soll auch die Dokumentation bzw. Hilfe wiederum in behindertengerechter Form darstellbar sein und natürlich auch die Zugangserleichternden Funktionen beinhalten, so z.B. eine Übersicht der Standardabkürzungen. Auch auf Änderungen zwischen verschiedenen Versionen soll aufmerksam gemacht werden.

2.5 Offene Fragen

Nach dem Vortrag kam es zu einer Frage: Wie sieht es mit Implementierungen aus? Gibt es UAAG 1.0 konforme Software?

Zunächst ist zu sagen, dass es drei Arten von Konformität mit der UAAG 1.0 gibt. Dabei spielt es eine Rolle, wie ein Benutzerprogramm die priorisierten Chckpoints einhält. Werden nur die mit Priorität 1 implementiert, spricht man von Konformitätsklasse A. Bei allen mit Priorität 1 und 2 von Konformitätsklasse AA und schließlich von AAA wenn alle Checkpoints implementiert sind.

Es gibt auf der Internetseite der UAAG Arbeitsgruppe der WAI eine Aufstellung von Benutzerprogrammen, die auf die Konformität getestet wurden. Es gibt unter diesen Programmen, unter anderem dem Internet Explorer, Mozilla, Opera und Konqueror, keins, dass in Klasse AAA liegt. Vielmehr gibt es in jeder Implementation noch eine Vielzahl von nicht erfüllten Checkpoints.

Quelle: Summary implementation report for UAAG 1.0, Web Access Initiative, W3C, http://www.w3.org/WAI/UA/impl-pr2/.

3 XML Events

Vorgetragen am 16.12.2004 von Ji-Hyun Lim und Oyunchimeg Chagnaadorj.

Der Vortrag unterteilte sich in folgende Teile:

- Die Motivation von XML Events
- Events in DOM2
- Events in HTML
- Das XML Events Modul
- Der Aufbau von XML Events

3.1 Motivation

In XML definierte Elemente sind statisch. Wünschenswert ist es allerdings den Elementen ein gewisses Verhalten zuzuweisen und somit einen Grad der Interaktion zwischen Elementen durch Reaktion auf Ereignisse zu realisieren. In diesem Zuge ist ein Ereignis eine asynchrone Erscheinung mit der ein Element mit einem Ziel in Bezug gesetzt wird. Da man mit DOM2 bereits eine mächtige Eventsteuerung standardisiert hat, wurde das XML Events Module definiert. Durch das XML Event Modul hat man eine generische Schnittstelle zwischen XML und seinen Dialekten auf der einen und DOM2 auf der anderen Seite zur Verfügung. Dadurch muß man nur dieses Modul benutzen, um mit Hilfe bestehender Standards Events in XML zu realisieren

3.2 Events in DOM2

Das Document Object Model (DOM) bildet XML Dokumente im Arbeitsspeicher als Baumstruktur ab. Über eine Programmschnittstelle gewährt der Standard den Zugriff auf Elemente und ihren Inhalt innerhalb dieser Baumstruktur. Somit kann man mit Programmier- oder Skriptsprachen, die diese Schnittstelle implementieren auf XML Dokumente zugreifen und bearbeiten. DOM gibt es inzwischen in zwei Leveln. Während DOM1 nur den Zugriff auf die reine Baumstruktur von HTML und XML Dokumenten erlaubt, gibt es in DOM2 zusätzlich Schnittstellen für den Zugriff auf CSS, die Navigation und evantbasierte Dokumente.

DOM2 Events haben die folgenden Eigenschaften:

- Es gibt ein einziges, einheitliches Ereignissystem
- Man kann Ereignislistener und -handler registrieren
- Ereignisse können durch die Baumstruktur weitergeleitet werden
- Zu jedem Ereignis kann man sich Kontextinformationen beschaffen
- Ein Ereignisfluß ist definiert

Dabei teilt sich der Ereignisfluß in zwei Teile auf. In der so genannten Capture Phase wird das Ereignis im Syntaxbaum von der Wurzel zum Zielelement weitergeleitet. In der zweiten Phase namens Bubbling Phase kehrt das Ereignis dann den Pfad wieder zurück, wobei in jedem Knoten, der als Observer fungiert, auf das Ereignis reagiert werden kann.

3.3 Events in HTML

Zum Vergleich mit XML Events werden HTML Events betrachtet. Da sie ebenfalls den Event Mechanismus von DOM2 benutzen, werden generell die gleichen Effekte generiert. Aber HTML Events sind eingeschränkter als XML Events. So haben Events fest spezifizierte Namen und die Eventbehandlung ist auch immer nur mit einer Skriptsprache möglich. Im Vergleich dazu kann man mit XML Events eigene Ereignistypen definieren und diese dann auch mit verschiedenen Handlern, die in beliebigen Sprachen implementiert sind, reagieren.

3.4 Das XML Events Modul

Wie einleitend erwähnt, stellt das XML Events Modul eine Schnittstelle zwischen XML und DOM dar. Dadurch soll erreicht werden, dass man Ereignisse in XML und seinen Abkömmlingen erstellen kann, ohne direkt was am jeweiligen XML Standard oder am DOM Standard ändern zu müssen. So führt die Recommendation auch keinen neuen Dokumenttyp ein, sondern stellt über einen Namensraum vielmehr ein neues Element bereit, um dies zu erreichen.

Das Element heißt listener und unterstützt die EventListener Schnittstelle des DOM. Mittels listener werden Ereignisse über EventListener deklariert und an Knoten in Syntaxbaum registriert. Folgende Attribute liefert listener:

- event - Der Ereignistyp, für den der Listener registriert wird
- observer - Die ID eines Elements für das der Listener registriert wird (optional)
- target - Die ID des Zielelements des Ereignisses (optional)
- handler - Eine URI, die den Speicherort des Ereignishandlers angibt (optional)
- phase - Angabe, wann der Listener innerhalb des DOM Ereignisflußes aktiviert wird (optional)
- propagate - Angabe, ob das Ereignis nach Abarbeitung der Listener eines Knoten weitertraversieren darf (optional)
- defaultAction - Angabe, ob nach Abarbeitung aller Listener eine optional voreingestellte Aktion ausgeführt werden soll (optional)
- id - Eine dokumentweit eindeutige ID zur Identifikation des Ereignisses. Verwendet, um im DOM das Element zu manipulieren (optional)

Bisher findet XML Events Verwendung in XForms und XHTML in Verbindung mit Voice. Außerdem ist XML Events in Vorbereitung in SVG 1.2 und XHTML 2 integriert zu sein.

3.5 Offene Fragen

Keine.