

XML w bazach danych, standardy wiążące dokumenty XML

Patryk Czarnik

Instytut Informatyki UW

XML i nowoczesne technologie zarządzania treścią – 2008/09

Walidacja względem DTD podczas parsowania

```
SAXParserFactory factory = SAXParserFactory.newInstance();  
factory.setValidating(true);  
XMLReader reader = factory.newSAXParser().getXMLReader();  
  
reader.setContentHandler(mojConentHandler);  
reader.setErrorHandler(mojErrorHandler);  
  
reader.parse(args[0]);
```

Walidacja względem XML Schema

```
SchemaFactory schemaFactory =
    SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
Schema schemat = schemaFactory.newSchema(new StreamSource(args[1]));

SAXParserFactory factory = SAXParserFactory.newInstance();
factory.setValidating(false);
factory.setSchema(schemat);
factory.setNamespaceAware(true);

XMLReader reader = factory.newSAXParser().getXMLReader();
reader.setContentHandler(mojConentHandler);
reader.setErrorHandler(mojErrorHandler);
reader.parse(args[0]);
```

Walidacja i zapis drzewa DOM

```
SchemaFactory schemaFactory =  
    SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);  
Schema schemat = schemaFactory.newSchema(new StreamSource(args[1]));  
Validator validator = schemat.newValidator();  
validator.validate(new DOMSource(doc));
```

```
DOMImplementationLS lsImpl =  
    (DOMImplementationLS)domImpl.getFeature("LS", "3.0");  
LSSerializer ser = lsImpl.createLSSerializer();  
LSOutput out = lsImpl.createLSOutput();  
out.setByteStream(new FileOutputStream(args[0]));  
ser.write(doc, out);
```

Walidacja i zapis drzewa DOM

```
SchemaFactory schemaFactory =
    SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
Schema schemat = schemaFactory.newSchema(new StreamSource(args[1]));
Validator validator = schemat.newValidator();
validator.validate(new DOMSource(doc));
```

```
DOMImplementationLS lsImpl =
    (DOMImplementationLS)domImpl.getFeature("LS", "3.0");
LSSerializer ser = lsImpl.createLSSerializer();
LSOutput out = lsImpl.createLSOutput();
out.setByteStream(new FileOutputStream(args[0]));
ser.write(doc, out);
```

Transformacje XSLT

```
TransformerFactory trans_fact = TransformerFactory.newInstance();
    transformer = trans_fact.newTransformer(new StreamSource(args[2]));

Source src = new StreamSource(args[0]);
Result res = new StreamResult(args[1]);

transformer.transform(src, res);
```

Transformacje

Zastosowanie do zapisu zdarzeń SAX po przefiltrowaniu

```
SAXParserFactory parser_fact = SAXParserFactory.newInstance();
XMLReader reader = parser_fact.newSAXParser().getXMLReader();

TransformerFactory trans_fact = TransformerFactory.newInstance();
Transformer transformer = trans_fact.newTransformer();

XMLFilter filtr = new FiltrGrupyWażne();
filtr.setParent(reader);

InputSource doc = new InputSource(args[0]);

Source src = new SAXSource(filtr, doc);
Result res = new StreamResult(args[1]);

transformer.transform(src, res);
```


Klasyfikacja wsparcia dla XML-a w bazach danych

- Baza danych ze wsparciem dla XML
(zwykle relacyjna, czasem „obiektoowa” bądź inna)
 - konfiguracja struktur danych jako tabel i powiązań,
 - eksport i import danych w postaci dokumentów XML,
 - struktura dokumentów XML pochodną relacyjnych struktur danych,
 - zastosowanie: integracja, wymiana danych;
- XML-owa baza danych:
 - konfiguracja struktur danych przy pomocy DTD/XML Schema,
 - wyszukiwanie z użyciem XQuery lub XPath,
 - indeksowanie elementów, atrybutów, wyrażeń XPath,
 - struktura danych – dokumenty XML,
 - możliwa optymalizacja wewnętrznego formatu danych,
 - zastosowanie: przechowywanie i przetwarzanie dokumentów strukturalnych.

Klasyfikacja wsparcia dla XML-a w bazach danych

- Baza danych ze wsparciem dla XML
(zwykle relacyjna, czasem „obiektowa” bądź inna)
 - konfiguracja struktur danych jako tabel i powiązań,
 - eksport i import danych w postaci dokumentów XML,
 - struktura dokumentów XML pochodną relacyjnych struktur danych,
 - zastosowanie: integracja, wymiana danych;
- XML-owa baza danych:
 - konfiguracja struktur danych przy pomocy DTD/XML Schema,
 - wyszukiwanie z użyciem XQuery lub XPath,
 - indeksowanie elementów, atrybutów, wyrażeń XPath,
 - struktura danych – dokumenty XML,
 - możliwa optymalizacja wewnętrznego formatu danych,
 - zastosowanie: przechowywanie i przetwarzanie dokumentów strukturalnych.

XML w relacyjnych bazy danych

- Korzyści:
 - integracja aplikacji, wymiana danych,
 - łatwe transformacje danych,
 - prezentacja danych.
- Problemy:
 - czy i jak przechowywać dokumenty XML w bazie danych?
 - metody dostępu (zadawania zapytań),
 - efektywność.

XML a relacyjne bazy danych

- Przechowywanie XML-a w relacyjnych bazach danych:
 - elementy dokumentu XML jako pola tabeli bazodanowej (dokument „rozłożony na czynniki pierwsze”),
 - dokument XML w całości przechowywany w polu bazy danych.
- Sposoby wspierania XML-a przez systemy zarządzania bazami danych:
 - generowanie XML-a na podstawie zawartości bazy danych,
 - wypełnianie zawartości bazy na podstawie zawartości dokumentu XML,
 - specjalne indeksowanie pól zawierających XML,
 - zapytania XPath/XQuery na danych typu XML,
 - wbudowane parsery XML i procesory XSLT.

XML a relacyjne bazy danych

- Przechowywanie XML-a w relacyjnych bazach danych:
 - elementy dokumentu XML jako pola tabeli bazodanowej (dokument „rozłożony na czynniki pierwsze”),
 - dokument XML w całości przechowywany w polu bazy danych.
- Sposoby wspierania XML-a przez systemy zarządzania bazami danych:
 - generowanie XML-a na podstawie zawartości bazy danych,
 - wypełnianie zawartości bazy na podstawie zawartości dokumentu XML,
 - specjalne indeksowanie pól zawierających XML,
 - zapytania XPath/XQuery na danych typu XML,
 - wbudowane parsery XML i procesory XSLT.

Przykład – XML w Oracle

- Wsparcie dla XML w Oracle 10g (<http://www.oracle.com/xml>).
- Parsery XML dostarczane przez Oracle:
 - pozwalają na wykorzystanie XML-a we własnych aplikacjach korzystających z bazy,
 - dostępne dla PL-SQL-a, Javy i C++.
- XML-SQL Utility:
 - generowanie XML-a bezpośrednio z bazy przy pomocy specjalnych zapytań,
 - wypełnianie bazy na podstawie zawartości dokumentu XML.
- Typ danych `XMLType`.

XML w Oracle – XML-SQL Utility

Eksport XML – funkcja `getXML()`

```
SELECT xmlgen.getXML('select * from emp') FROM dual;
```

```
<rowset>
  <row id="1">
    <empno>10</empno>
    <name>Scott Tiger</name>
    <title>specialist</title>
  </row>
  ...
</rowset>
```

XML w Oracle – XMLType

- XMLType – specjalny typ danych:
 - kolumny, tabele, perspektywy, zmienne, ...
 - indeksowanie zawartości XML,
 - zapytania XQuery,
 - kontrola poprawności strukturalnej względem XML Schema,
 - przekształcenia XSLT.
- Specjalne operatory:
 - `extract`, `extractValue`, `existsNode`,
`transform`, `updateXML`, `XMLSequence`.
- *XPath Rewrite* – przekształcanie ścieżek XPath w równoważne konstrukcje SQL na wewnętrznej reprezentacji strukturalnej XMLType.

Bazy danych ze wsparciem dla XML

Istotne wsparcie

- DB2, IBM (wersja 9 – *pureXML*)
- Oracle (od 8i)
- Microsoft SQL Server (od wersji 2000)
- Sybase ASE 12.5

Minimalne wsparcie

- MySQL (zapytania XPath nad węzłami tekstowo zawierającymi XML)
- PostgreSQL (walidacja i ścieżki XPath, interfejs ma być zmieniony)

Bazy danych ze wsparciem dla XML

Istotne wsparcie

- DB2, IBM (wersja 9 – *pureXML*)
- Oracle (od 8i)
- Microsoft SQL Server (od wersji 2000)
- Sybase ASE 12.5

Minimalne wsparcie

- MySQL (zapytania XPath nad węzłami tekstowo zawierającymi XML)
- PostgreSQL (walidacja i ścieżki XPath, interfejs ma być zmieniony)

XML-owa baza danych

- Warstwa logiczna:
 - dokument XML jako podstawowa jednostka przechowywanych danych,
 - schemat jako definicja struktury,
 - kolekcje dokumentów.
- Warstwa fizyczna:
 - niekoniecznie tekstowo zapisane dokumenty XML,
 - indeksy.
- Funkcjonalności typowe dla baz danych:
 - interfejs do aktualizacji danych,
 - transakcje i współbieżny dostęp,
 - bezpieczeństwo.

XML-owa baza danych

- Warstwa logiczna:
 - dokument XML jako podstawowa jednostka przechowywanych danych,
 - schemat jako definicja struktury,
 - kolekcje dokumentów.
- Warstwa fizyczna:
 - niekoniecznie tekstowo zapisane dokumenty XML,
 - indeksy.
- Funkcjonalności typowe dla baz danych:
 - interfejs do aktualizacji danych,
 - transakcje i współbieżny dostęp,
 - bezpieczeństwo.

XML-owa baza danych

- Warstwa logiczna:
 - dokument XML jako podstawowa jednostka przechowywanych danych,
 - schemat jako definicja struktury,
 - kolekcje dokumentów.
- Warstwa fizyczna:
 - niekoniecznie tekstowo zapisane dokumenty XML,
 - indeksy.
- Funkcjonalności typowe dla baz danych:
 - interfejs do aktualizacji danych,
 - transakcje i współbieżny dostęp,
 - bezpieczeństwo.

Tamino – XML-owa baza danych

- Pierwszy serwer „bazodanowy” przechowujący dane w XML-u.
- Komunikacja:
 - za pośrednictwem protokołu HTTP, bezpośrednio przez URL,
 - moduł X-Node, zapewniający integrację z innymi źródłami danych:
 - ODBC, OLE DB,
 - system plików.
- Platforma dla:
 - aplikacji internetowych typu B2C,
 - elektronicznej wymiany dokumentów:
 - wsparcie dla XML Signature;
 - systemów zarządzania treścią:
 - wersjonowanie, scalanie, indeksowanie dokumentów nie-XML.

Tamino – XML-owa baza danych

- Pierwszy serwer „bazodanowy” przechowujący dane w XML-u.
- Komunikacja:
 - za pośrednictwem protokołu HTTP, bezpośrednio przez URL,
 - moduł X-Node, zapewniający integrację z innymi źródłami danych:
 - ODBC, OLE DB,
 - system plików.
- Platforma dla:
 - aplikacji internetowych typu B2C,
 - elektronicznej wymiany dokumentów:
 - wsparcie dla XML Signature;
 - systemów zarządzania treścią:
 - wersjonowanie, scalanie, indeksowanie dokumentów nie-XML.

XML:DB

- Inicjatywa specyfikacji XML-owych baz danych.
- *XML Database API (XAPI)*
 - kolekcje zasobów, „zasób” odpowiada dokumentowi XML,
 - odczyt i zapis (całego) zasobu poprzez interfejsy DOM i SAX
 - usługi: zapytania XPath, transakcje, zarządzanie kolekcjami,
 - ostatnia wersja: wrzesień 2001.
- *XML Update Language (XUpdate)*
 - język – zastosowanie XML,
 - wstawianie, usuwanie i aktualizacja „rekordów” danych,
 - adresowanie ścieżkami XPath,
 - ostatnia wersja: wrzesień 2000.

XML:DB

- Inicjatywa specyfikacji XML-owych baz danych.
- *XML Database API (XAPI)*
 - kolekcje zasobów, „zasób” odpowiada dokumentowi XML,
 - odczyt i zapis (całego) zasobu poprzez interfejsy DOM i SAX
 - usługi: zapytania XPath, transakcje, zarządzanie kolekcjami,
 - ostatnia wersja: wrzesień 2001.
- *XML Update Language (XUpdate)*
 - język – zastosowanie XML,
 - wstawianie, usuwanie i aktualizacja „rekordów” danych,
 - adresowanie ścieżkami XPath,
 - ostatnia wersja: wrzesień 2000.

XUpdate – przykład

Przykład z dokumentacji standardu

```
<?xml version="1.0"?>
<xupdate:modifications version="1.0"
    xmlns:xupdate="http://www.xmldb.org/xupdate">

  <xupdate:insert-after select="/addresses/address[1]" >

    <xupdate:element name="address">
      <xupdate:attribute name="id">2</xupdate:attribute>
      <fullname>Lars Martin</fullname>
      <born day='2' month='12' year='1974' />
      <town>Leizig</town>
      <country>Germany</country>
    </xupdate:element>

  </xupdate:insert-after>
</xupdate:modifications>
```

XML-owe bazy danych – przegląd

produkt	licencja	zapytania	XML:DB API
Apache XIndex	open source	XPath	tak
BaseX	open source	XPath, XQuery	tak
eXist	open source	XPath, XQuery	częściowo
Sedna	open source	XPath, XQuery	tak
Tamino	płatny	XQuery, XPath	częściowo
Gemfire Enterprise	płatny	XQuery, OQL	tak
DOMSafeXML (?)	płatny	XQuery, XPath	tak?

źródło: *Wikipedia i strony WWW produktów*

XPointer

- Adresowanie dokumentów XML i ich fragmentów, zgodnie ze składnią URI:

`http://www.sejm.gov.pl/ustawa.xml#def-podatnik`

- Rekomendacje W3C z 25 marca 2003:
 - XPointer Framework (zawiera schemat `xpointer()`),
 - XPointer `element()` scheme,
 - XPointer `xmlns()` scheme.

Schematy xpointer i xmlns

xpointer

- Ścieżki XPath

```
http://www.sejm.gov.pl/ustawa.xml#xpointer(/art[5]/par[2])
```

xmlns

- Obsługa przestrzeni nazw
- Do wykorzystania w dalszej części ścieżki

```
ustawa.xml#xmlns(pr=http://xxx/prawo)xpointer(/art[5]/par[2])
```

Schematy xpointer i xmlns

xpointer

- Ścieżki XPath

```
http://www.sejm.gov.pl/ustawa.xml#xpointer(/art[5]/par[2])
```

xmlns

- Obsługa przestrzeni nazw
- Do wykorzystania w dalszej części ścieżki

```
ustawa.xml#xmlns(pr=http://xxx/prawo)xpointer(/art[5]/par[2])
```

XPointer element() scheme

- Adresowanie elementów po wartościach atrybutu typu ID:

```
jakiś.xml#element (def-podatnik)
```

- Adresowanie ze względu na pozycję:

```
jakiś.xml#element (/1/4/3)
```

```
jakiś.xml#element (def-podatnik/2/3)
```

- Składnia skrócona:

```
jakiś.xml#def-podatnik
```

```
jakiś.xml#/1/2/3
```

```
jakiś.xml#def-podatnik/2/3
```

XPointer element() scheme

- Adresowanie elementów po wartościach atrybutu typu ID:

```
jakiś.xml#element (def-podatnik)
```

- Adresowanie ze względu na pozycję:

```
jakiś.xml#element (/1/4/3)
```

```
jakiś.xml#element (def-podatnik/2/3)
```

- Składnia skrócona:

```
jakiś.xml#def-podatnik
```

```
jakiś.xml#/1/2/3
```

```
jakiś.xml#def-podatnik/2/3
```

XPointer element() scheme

- Adresowanie elementów po wartościach atrybutu typu ID:

```
jakiś.xml#element (def-podatnik)
```

- Adresowanie ze względu na pozycję:

```
jakiś.xml#element (/1/4/3)
```

```
jakiś.xml#element (def-podatnik/2/3)
```

- Składnia skrócona:

```
jakiś.xml#def-podatnik
```

```
jakiś.xml#/1/2/3
```

```
jakiś.xml#def-podatnik/2/3
```

XInclude – włączanie zawartości dokumentów

- Załączanie zawartości jednego dokumentu XML do innego.
- Znaczniki w warstwie logicznej (elementy XML), nie fizycznej (jak referencje do encji).
- Możliwości:
 - załączanie całych dokumentów (nazwa pliku lub URL),
 - załączanie fragmentów (wskazanych przez XPointer),
 - określanie zawartości używanej w razie błędu.
- Przezroczysta dla czytelnika interpretacja XInclude przez parser (m.in. jako opcja w JAXP 1.4).

XInclude – włączanie zawartości dokumentów

- Załączanie zawartości jednego dokumentu XML do innego.
- Znaczniki w warstwie logicznej (elementy XML), nie fizycznej (jak referencje do encji).
- Możliwości:
 - załączanie całych dokumentów (nazwa pliku lub URL),
 - załączanie fragmentów (wskazanych przez XPointer),
 - określanie zawartości używanej w razie błędu.
- Przezroczysta dla czytelnika interpretacja XInclude przez parser (m.in. jako opcja w JAXP 1.4).

XInclude – włączanie zawartości dokumentów

- Załączanie zawartości jednego dokumentu XML do innego.
- Znaczniki w warstwie logicznej (elementy XML), nie fizycznej (jak referencje do encji).
- Możliwości:
 - załączanie całych dokumentów (nazwa pliku lub URL),
 - załączanie fragmentów (wskazanych przez XPointer),
 - określanie zawartości używanej w razie błędu.
- Przezroczysta dla czytelnika interpretacja XInclude przez parser (m.in. jako opcja w JAXP 1.4).

XInclude – przykład

Przykład: źródło

```
<wynik>
  <xi:include xmlns:xi="http://www.w3.org/2001/XInclude"
    href="salatka.xml#xpointer(/przepis/tytul)">
    <xi:fallback><błąd>Brak przepisu</błąd></xi:fallback>
  </xi:include>
</wynik>
```

Przykład: po przetworzeniu

```
<wynik>
  <tytul>Sałatka z ogórków</tytul>
</wynik>
```

XInclude – przykład

Przykład: źródło

```
<wynik>
  <xi:include xmlns:xi="http://www.w3.org/2001/XInclude"
    href="salatka.xml#xpointer(/przepis/tytul)">
    <xi:fallback><błąd>Brak przepisu</błąd></xi:fallback>
  </xi:include>
</wynik>
```

Przykład: po przetworzeniu

```
<wynik>
  <tytul>Sałatka z ogórków</tytul>
</wynik>
```

XLink – dowiązania w XML-u

- **Odknośniki znane z HTML:**
 - łączą dwa dokumenty: źródło i cel linku,
 - źródłem linku jest zawsze element opisujący link (A, IMG).
- **XLink – rozszerzona koncepcja dowiązań:**
 - dowolne elementy przechowują informacje o linkach,
 - nieistotna nazwa elementu,
 - istotne atrybuty z przestrzeni nazw XLink,
 - więcej niż dwa końce linku,
 - możliwość opisanie powiązania poza wiązanyimi dokumentami.
- **Status:**
 - korzenie historyczne: HyTime,
 - XLink 1.0 – rekomendacja W3C, czerwiec 2001,
 - XLink 1.1 – wersja robocza (aktualizacja w 2008).

XLink – dowiązania w XML-u

- Odnośniki znane z HTML:
 - łączą dwa dokumenty: źródło i cel linku,
 - źródłem linku jest zawsze element opisujący link (A, IMG).
- XLink – rozszerzona koncepcja dowiązań:
 - dowolne elementy przechowują informacje o linkach,
 - nieistotna nazwa elementu,
 - istotne atrybuty z przestrzeni nazw XLink,
 - więcej niż dwa końce linku,
 - możliwość opisania powiązania poza wiązanyymi dokumentami.
- Status:
 - korzenie historyczne: HyTime,
 - XLink 1.0 – rekomendacja W3C, czerwiec 2001,
 - XLink 1.1 – wersja robocza (aktualizacja w 2008).

XLink – dowiązania w XML-u

- Odnośniki znane z HTML:
 - łączą dwa dokumenty: źródło i cel linku,
 - źródłem linku jest zawsze element opisujący link (A, IMG).
- XLink – rozszerzona koncepcja dowiązań:
 - dowolne elementy przechowują informacje o linkach,
 - nieistotna nazwa elementu,
 - istotne atrybuty z przestrzeni nazw XLink,
 - więcej niż dwa końce linku,
 - możliwość opisania powiązania poza wiązanyimi dokumentami.
- Status:
 - korzenie historyczne: HyTime,
 - XLink 1.0 – rekomendacja W3C, czerwiec 2001,
 - XLink 1.1 – wersja robocza (aktualizacja w 2008).

Terminologia

- **Zasób** (*resource*) – dowolna adresowalna jednostka informacji lub usługa.
- **Dowiązanie** (*link*) – jawnie wyrażona (przy pomocy **elementu wiążącego** (*linking element*)) relacja pomiędzy zasobami.
 - te zasoby **uczestniczą** (*participate*) w dowiązaniu.
- **Przejście** (*traversal*) – użycie pary zasobów połączonej dowiązaniem.
- **Łuk** (*arc*) – informacja o przejściu między dwoma zasobami (kierunek, zachowanie aplikacji, itp.):
 - **wychodzący** (*outbound*),
 - **wchodzący** (*inbound*),
 - **niezależny** (*third party*).

Dowiązania XLink

Extended link

- wiąże dowolną liczbę zasobów:
 - zasoby zewnętrzne (np. odwołania do innych dokumentów),
 - zasoby lokalne (zawarte w elemencie wiążącym).
- łuki opisujące sposoby przechodzenia pomiędzy zasobami,
- role zasobów uczestniczących w linku,
- role łuków.

Simple link

- link wychodzący,
- wiąże dokładnie 2 zasoby: 1 lokalny i 1 zewnętrzny,
- jeden łuk z zasobu lokalnego do zewnętrznego.

Dowiązania XLink

Extended link

- wiąże dowolną liczbę zasobów:
 - zasoby zewnętrzne (np. odwołania do innych dokumentów),
 - zasoby lokalne (zawarte w elemencie wiążącym).
- łuki opisujące sposoby przechodzenia pomiędzy zasobami,
- role zasobów uczestniczących w linku,
- role łuków.

Simple link

- link wychodzący,
- wiąże dokładnie 2 zasoby: 1 lokalny i 1 zewnętrzny,
- jeden łuk z zasobu lokalnego do zewnętrznego.

Simple link – przykład

```
<osoba xmlns:xlink="http://www.w3.org/1999/xlink">
<nazwisko>Kopernik, Mikołaj</nazwisko>
<biogram>Wybitny polski astronom, matematyk, lekarz,
prawnik, tłumacz poezji włoskiej i ekonomista, pochodził
z rodziny wywodzącej się z mieszczan krakowskich.
Urodzony w
  <geogr xlink:type="simple" xlink:href="Torun.xml">Toruniu</geogr>
.</biogram>
</osoba>
```

Simple link – przykład wykorzystujący XPointer

```
<osoba xmlns:xlink="http://www.w3.org/1999/xlink">
<nazwisko>Kopernik, Mikołaj</nazwisko>
<biogram>Wybitny polski astronom, matematyk, lekarz,
prawnik, tłumacz poezji włoskiej i ekonomista, pochodził
z rodziny wywodzącej się z mieszczan krakowskich.
Urodzony w
  <geogr xlink:type="simple"
        xlink:href="encyklopedia.xml#Toruń">Toruniu</geogr>
.</biogram>
</osoba>
```

Extended link – przykład

```
<fikcja xmlns:xlink="http://www.w3.org/1999/xlink"
  xlink:type="extended">

  <wypowiedz xlink:type="resource">Kopernik była
    kobieta!</wypowiedz>

  <film      xlink:type="locator"
            xlink:href="seksmisja.xml"
            xlink:title="Seksmisja"/>

  <osoba     xlink:type="locator"
            xlink:href="kopernik.xml"
            xlink:title="Kopernik, Mikołaj"/>

  <pojecie  xlink:type="locator"
            xlink:href="kobieta.xml"
            xlink:title="kobieta"/>

</fikcja>
```

Atrybuty w linkach rozszerzonych

`type` rola elementu w linku

`href` adres zasobu zewnętrznego

`role` abstrakcyjny identyfikator roli zasobu w powiązaniu (URI)

`arcrole` j.w. ale dla pojedynczego łuku między zasobami

`title` etykieta tekstowa zasobu, łuku itd.

`show` jak prezentować

`actuate` kiedy aktywować

`label` etykieta zasobu używana w definicjach łuków

`from, to` etykieta zasobu używana w definicjach łuków

Wartości wybranych atrybutów

- **xlink:type**

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

- **xlink:show**

- embed, new, replace,
- other, none

- **xlink:actuate**

- onLoad, onRequest,
- other, none

Dopuszczalność atrybutów w zależności od typu

	simple	extended	locator	arc	resource	title
type	R	R	R	R	R	R
href	O		R			
role	O	O	O		O	
arcrole	O			O		
title	O	O	O	O	O	
show	O			O		
actuate	O			O		
label			O		O	
from				O		
to				O		

Przyszłość XLink

- Zastosowania:
 - organizowanie, kojarzenie zasobów, nawet gdy nie mamy prawa zapisu,
 - dostarczanie wartości dodanej – zbiorów linków.
- Zasięg:
 - lokalny – serwery linków operujące na bazie linków,
 - Internet?
- Problemy:
 - wizualizacja *extended links*,
 - synchronizacja zasobów i linków (Internet).