

XML w elektronicznej wymianie danych, integracji aplikacji i bezpieczeństwie

Patryk Czarnik

Instytut Informatyki UW

XML i nowoczesne technologie zarządzania treścią – 2008/09

Elektroniczna wymiana danych (EDI) – motywacja

- ▶ Możliwości wymiany danych między firmami/institucjami (B2B):
 - ▶ dokumenty na papierze,
 - ▶ elektroniczna wymiana danych.
- ▶ Możliwości ustalenia protokołu EDI:
 - ▶ dostawca (sprze)daje klientowi narzędzia,
 - ▶ partner dostosowuje się do formatu silniejszego partnera,
 - ▶ tworzone ad-hoc narzędzia tłumaczące,
 - ▶ standard.
- ▶ Możliwości wykorzystania standardu:
 - ▶ całe oprogramowanie tworzone zgodnie ze standardem,
 - ▶ wewnętrzny format odmienny, narzędzia tłumaczące do standardu w interfejsach
 - ▶ możliwość adaptacji istniejącego systemu.

Standaryzacja EDI

- ▶ Standard EDI powinien określać:
 - ▶ ogólny format (składnię) komunikatów,
 - ▶ sposób definiowania schematów komunikatów (schemat – lista pól komunikatów z ich typami danych, rozmiarem itp., czasem możliwość zagnieżdżania struktur).
- ▶ Standard EDI zwykle nie określa (ewentualnie określone dla konkretnego zastosowania):
 - ▶ sposobu transmisji danych,
 - ▶ pól zawartych w konkretnych komunikatach i ich znaczenia.

Standaryzacja EDI przed XML

- ▶ ANSI Accredited Standards Committee X12 sub-group
 - ▶ standard narodowy USA – Ameryka Płn.
- ▶ EDIFACT
 - ▶ standard światowy – Europa i Azja,
 - ▶ składnia komunikatów,
 - ▶ słownik standardowych **elementów**,
 - ▶ protokół komunikacji (ale można korzystać z innych).

EDIFACT

EDIFACT – przykład komunikatu

```
UNB+IATB:1+6XPPC+LHPPC+940101:0950+1'  
UNH+1+PAORES:93:1:IA'  
MSG+1:45'  
IFT+3+XYZCOMPANY AVAILABILITY'  
ERC+A7V:1:AMD'  
IFT+3+NO MORE FLIGHTS'  
ODI'  
TVL+240493:1000::1220+FRA+JFK+DL+400+C'  
PDI++C:3+Y::3+F::1'  
APD+74C:0::6+++++6X'  
TVL+240493:1740::2030+JFK+MIA+DL+081+C'  
PDI++C:4'  
APD+EM2:0:1630::6+++++DA'  
UNT+13+1'  
UNZ+1+1'
```

EDIFACT – struktura

Hierarchia

1. wymiana (*interchange*)
2. komunikat (*message*)
3. grupa – opcjonalnie, może zawierać inne grupy
4. segment (separator: ')
5. element złożony (*composite*, separator: +)
6. element z danymi (separator: :)

Schematy

- ▶ składnia do zapisywania schematów komunikatów
- ▶ definiowanie komunikatów, grup, segmentów, elementów
- ▶ możliwość określenia opcjonalności i powtórzeń grup i segmentów

XML EDI

Idea: zastosowanie XML jako formatu danych EDI.

Tradycyjne EDI

- ▶ Format dokumentów zapisany w specyfikacji
- ▶ Zwięzłe komunikaty, tylko niezbędne dane.
- ▶ Scentralizowana, trudna zmiana standardu.
- ▶ Zmiany standardu pociągają uciążliwe zmiany oprogramowania.
- ▶ Implementowanie od podstaw.

XML EDI

- ▶ „Samoopisujący się” format dokumentów.
- ▶ Rozwlekłe komunikaty – narzut na „samoopisywanie się”.
- ▶ Możliwość tworzenia własnych odmian standardów.
- ▶ Większość problemów ze zmianą standardu bierze na siebie parser XML.
- ▶ Możliwość korzystania z gotowych narzędzi.

Elastyczność XML EDI

- ▶ Elastyczny format komunikatów (elementy opcjonalne, wybór podelementu).
- ▶ XSLT jako język opisu formatowania.
- ▶ Nowe zastosowania EDI:
 - ▶ podstawowa funkcjonalność – wymiana danych między aplikacjami przedsiębiorstw,
 - ▶ nowe perspektywy: kontakt z klientami wyposażonymi tylko w przeglądarki,
 - ▶ E-Commerce.
- ▶ Zgodność z ideą Web Services.

Standaryzacja XML EDI

- ▶ XML zbyt elastyczny.
- ▶ Standaryzacja na poziomie ramowym:
 - ▶ możliwa wymiana informacji dowolnego typu,
 - ▶ informacje jednego typu tak samo reprezentowane,
 - ▶ przykład: Electronic Business XML – ebXML.
- ▶ Standardy branżowe (konkretne zestawy komunikatów):
 - ▶ SWIFT,
 - ▶ RosettaNet,
 - ▶ Automotive Industry Action Group,
 - ▶ Health Level Seven,
 - ▶ Open Travel Alliance,
 - ▶ ...

ebXML

- ▶ ebXML:
 - ▶ zbiór specyfikacji definiujących sposób prowadzenia biznesu i wymiany danych przez Internet,
 - ▶ zaakceptowane 14 maja 2001 r.,
 - ▶ oczekiwane implementacje i wsparcie w istniejących systemach,
 - ▶ wsparcie przez inne inicjatywy standaryzacyjne.
- ▶ Electronic Business XML Working Group:
 - ▶ założona we wrześniu 1999 r.,
 - ▶ ok. 150 specjalistów,
 - ▶ patronat OASIS i UN/CEFACT.

ebXML – podejście do standaryzacji

- ▶ Meta-model pozwalający na opracowywanie modeli specyficznych dla zastosowań:
 - ▶ zbiór podstawowych schematów, elementów XML oraz procesów biznesowych,
 - ▶ sposób definiowania słowników danych,
 - ▶ nie definiuje konkretnych, docelowych komunikatów.
- ▶ Metainformacje:
 - ▶ informacje o wersjach,
 - ▶ metadane odpowiadające nagłówkom z istniejących systemów EDI.
- ▶ Ramy architektury technicznej:
 - ▶ sposoby implementacji repozytoriów, serwisów, itp.,
 - ▶ integracja z istniejącymi technologiami EDI.

XML w integracji aplikacji

- ▶ Cel: umożliwienie wymiany danych pomiędzy aplikacjami:
 - ▶ aplikacje/komponenty/moduły posługują się różnymi formatami wewnętrznymi,
 - ▶ wspólny mianownik: XML.
- ▶ Zastosowania:
 - ▶ komunikacja między klientem a serwerem,
 - ▶ komunikacja między elementami systemu rozproszonego,
 - ▶ integracja komponentów aplikacji,
 - ▶ konfigurowanie aplikacji i jej komponentów,
 - ▶ ...

Zastosowania

- ▶ Lokalne:
 - ▶ w ramach jednego projektu / jednej instytucji,
 - ▶ komunikacja między różnymi modułami / aplikacjami,
 - ▶ być może w architekturze rozproszonej,
 - ▶ rozwiązania tworzone pod kątem projektu,
 - ▶ możliwość stosowania rozwiązań standardowych (np. JAXB).
- ▶ Globalne:
 - ▶ usługi dostępne w Internecie,
 - ▶ współpraca różnych partnerów,
 - ▶ ważna standaryzacja,
 - ▶ standard – Web Services.

Web Services

- ▶ Idea – „witryna WWW dla programów”.
- ▶ Zazwyczaj dotyczy systemów o poniższych cechach (w nawiasach typowe/standardowe technologie):
 - ▶ komunikacja w wysokiej warstwie sieci (HTTP),
 - ▶ węzły/usługi opisane (WSDL),
 - ▶ komunikaty między węzłami (XML, SOAP),
 - ▶ możliwość rejestrowania i wyszukiwania usług (UDDI).

Web Services – możliwe zastosowania

- ▶ Udostępnianie/sprzedaż użytecznych danych:
 - ▶ rozkłady lotów linii lotniczych,
 - ▶ dane o pogodzie,
 - ▶ aktualny czas wg zegara atomowego.
- ▶ Zdalne usługi:
 - ▶ wyszukiwanie,
 - ▶ pobieranie aktualnej wersji oprogramowania.
- ▶ Operacje biznesowe między partnerami:
 - ▶ rezerwacja biletów, miejsc w hotelach itp,
 - ▶ zamawianie,
 - ▶ sprawdzanie stanu realizacji zamówienia,
 - ▶ elektroniczna wymiana danych.

Web Services – standaryzacja

- ▶ SOAP (pierwotnie *Simple Object Access Protocol*):
 - ▶ początki: 1998,
 - ▶ wersja 1.2: rekomendacja W3C, czerwiec 2003 (errata: kwiecień 2007).
- ▶ Web Services Description Language:
 - ▶ wersja 1.1: notatka W3C, 2001,
 - ▶ wersja 2.0: rekomendacja W3C, czerwiec 2007.
- ▶ Universal Description Discovery and Integration:
 - ▶ projekt organizacji OASIS,
 - ▶ część standardu WS-I Basic Profile.
- ▶ Standardy WS-*:
 - ▶ różne standardy, zazwyczaj nie rekomendacje W3C, m.in.:
 - ▶ Web Services Interoperability – ramy dla działania Web serwisów:
WS-I Basic Profile, Simple Soap Binding Profile, ... ,
 - ▶ WS-Eventing, WS-Addressing, WS-Routing, ... – standardy IBM.
- ▶ Business Process Execution Language (OASIS) – opis semantyki i łączenie serwisów w super-serwisy.

SOAP – protokół komunikacji

- ▶ Protokół komunikacji.
- ▶ Format komunikatów (zastosowanie XML).
- ▶ Różnice w stosunku do RPC, CORBA, DCOM itp.:
 - ▶ reprezentacja danych niezależna od platformy (tekstowa),
 - ▶ typy danych niezależne od platformy,
 - ▶ niższa efektywność.

Komunikat SOAP

- ▶ Dokument XML, jedna wiadomość:
 - ▶ przestrzeń nazw
`http://www.w3.org/2001/12/soap-envelope`,
 - ▶ element główny: `Envelope`.
- ▶ Składniki dokumentu (elementy pod głównym):
 - ▶ `header` – opcjonalny,
 - ▶ `body` – obowiązkowy.
- ▶ Ograniczenia:
 - ▶ brak DTD (i referencji do encji),
 - ▶ brak instrukcji przetwarzania.

Nagłówek SOAP

- ▶ `actor` – opcjonalny identyfikator strony komunikacji (URI),
- ▶ `mustUnderstand` – czy zrozumienie jest konieczne (0/1).

Przykład z W3Schools

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Header>
    <m:Trans
xmlns:m="http://www.w3schools.com/transaction/"
soap:actor="http://www.w3schools.com/appml/"
soap:mustUnderstand="1">234</m:Trans>
  </soap:Header>

  ...
</soap:Envelope>
```

Ciało komunikatu SOAP

- ▶ `encodingStyle` – sposób kodowania danych (URI),

Zapytanie – przykład z W3Schools

```
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Body>
    <m:GetPrice xmlns:m="http://www.w3schools.com/prices">
      <m:Item>Apples</m:Item>
    </m:GetPrice>
  </soap:Body>

</soap:Envelope>
```

Ciało komunikatu SOAP

Odpowiedź – przykład z W3Schools

```
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Body>
    <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
      <m:Price>1.90</m:Price>
    </m:GetPriceResponse>
  </soap:Body>

</soap:Envelope>
```

SOAP – odpowiedź informująca o błędzie

Przykład

```
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body>
  <soap:Fault>
    <soap:faultcode>Receiver</soap:faultcode>
    <soap:faultstring>Brak danych</soap:faultstring>
    <soap:faultdetail>Nie znaleziono studenta o numerze indeksu 123</soap:
  </soap:Fault>
</soap:Body>

</soap:Envelope>
```

Informacja zwrotna o błędzie

- ▶ Element(y) fault w body.
- ▶ Podelementy:
 - ▶ faultcode
 - ▶ faultstring
 - ▶ faultactor
 - ▶ detail
- ▶ Standardowe kody błędów:
 - ▶ VersionMismatch
 - ▶ MustUnderstand
 - ▶ DataEncodingUnknown
 - ▶ Sender
 - ▶ Receiver

WSDL – opis serwisu

- ▶ Dokument XML, opis jednej lub kilku usług:
 - ▶ przestrzeń nazw `http://schemas.xmlsoap.org/wsdl/`,
 - ▶ element główny: `definitions`.
- ▶ Składniki dokumentu (elementy pod głównym):
 - ▶ `serviceType` – opis usługi, zawiera wiele `portType`,
 - ▶ `portType` – typ „portu”, czyli zbiór operacji, usługa może posiadać wiele portów, `portType` może występować w `serviceType` oraz bezpośrednio w `definitions`,
 - ▶ `message` – opis typu komunikatu,
 - ▶ `types` – definicje typów,
 - ▶ `binding` – opis sposobu komunikacji.
- ▶ XML Schema używane do definiowania typów.

WSDL – wiadomości, typ portu

Przykład z W3Schools

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

WSDL – wiązanie (z SOAP)

- ▶ `style` – `rpc` lub `document`,
- ▶ `transport` – protokół transportowy (URI).
- ▶ `soapAction` – akcja SOAP odpowiadająca operacji.

Przykład z W3Schools

```
<binding type="glossaryTerms" name="b1">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http" />
  <operation>
    <soap:operation
      soapAction="http://example.com/getTerm"/>
    <input> <soap:body use="literal"/> </input>
    <output> <soap:body use="literal"/> </output>
  </operation>
</binding>
```

Wyszukiwanie usług

- ▶ Idea:
 - ▶ dostawcy usług rejestrują je w katalogu,
 - ▶ użytkownik znajduje usługę w katalogu,
 - ▶ najlepiej: opis usługi i wyszukiwanie semantyczne.
- ▶ Universal Description Discovery and Integration (UDDI):
 - ▶ usługa katalogowa, wiele powiązanych węzłów,
 - ▶ dostępna jako Web service (poprzez SOAP),
 - ▶ opisy usług w WSDL.

Super usługi i kooperacja usług

- ▶ Terminologia:
 - ▶ **choreografia** (*choreography*) – opis działania układu usług jako całości,
 - ▶ **orkiestracja** (*orchestration*) – opis działania pojedynczego komponentu i jego kooperacji ze światem,
- ▶ Standaryzacja:
 - ▶ WS-Choreography (W3C),
 - ▶ Business Process Execution Language (BPEL, OASIS) – opis orkiestracji pojedynczej usługi,
- ▶ (Nowe) usługi zbudowane w oparciu o istniejące usługi:
 - ▶ możliwość rekursji,
 - ▶ *Service Oriented Architecture*.

AJAX

- ▶ **AJAX – Asynchronous JavaScript and XML:**
 - ▶ strona WWW „działa” w przeglądarce użytkownika dzięki JavaScript,
 - ▶ komunikacja z serwerem po HTTP,
 - ▶ komunikaty kodowane w XML,
 - ▶ coraz szerzej stosowane (m.in. wiele usług Google).
- ▶ **Zalety:**
 - ▶ interakcja z użytkownikiem bez przeładowywania strony,
 - ▶ przesyłane tylko niezbędne dane.
- ▶ **Wady:**
 - ▶ narzut na przesłanie i uruchomienie skryptu,
 - ▶ (czasami) nieintuicyjne zachowanie stron WWW.
- ▶ **Odstępstwa w praktyce:**
 - ▶ inne „scripty”,
 - ▶ dane nie zawsze w XML.

XML w bezpieczeństwie

Za Pawłem Radzińskim

Najważniejsze aspekty bezpieczeństwa

poufność realizacja: szyfrowanie

uwierzytelnienie realizacja: podpis elektroniczny

Standardy XML związane z bezpieczeństwem

- ▶ Podpisy – XML Signature.
- ▶ Szyfrowanie – XML Encryption.
- ▶ Zastosowanie: m.in. w Web Services.

XML Signature

- ▶ Podpis dokumentu XML-owego zapisany w postaci struktury XML-owej.
- ▶ Podpis umieszczany w elemencie `Signature`:
 - ▶ w osobnym dokumencie (*detached signature*),
 - ▶ dołączonym do podpisywanego dokumentu (*enveloped signature*),
 - ▶ zawierającym podpisywane dane (*enveloping signature*).
- ▶ Możliwości XML Signature:
 - ▶ podpisywanie fragmentów dokumentu XML,
 - ▶ podpisywanie zasobów zewnętrznych (dostępnych poprzez URL),
 - ▶ podpisy wielokrotne.

XML Signature – przykład 1 (*detached*)

```
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm=
      "http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm=
      "http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
    <!-- w URI znajduje się wskazanie na podpisywane dane - tu zewnętrzne -->
    <Reference URI="http://przyklad.pl/pliki/do-podpisu.xml">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#base64"/>
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>60NvZvtdTB+7UnlLp/H24p7h4bs=</DigestValue>
    </Reference>
  </SignedInfo>
  <!-- zaszyfrowany skrót z SignedInfo - podpis -->
  <SignatureValue>OsH9A1jTNL...</SignatureValue>
  <KeyInfo><KeyValue><DSAKeyValue>
    <P>imup6lm...</P><Q>xDve3j7...</Q><G>NlugAf...</G>
    <Y>W7dOmH/v...</Y>
  </DSAKeyValue></KeyValue></KeyInfo>
</Signature>
```

Źródło: Kazienko, P., Co tam panie w XML-u?, Software 2.0, 6/2003

XML Signature – przykład 2 (*enveloped*)

```
<?xml version="1.0" encoding="UTF-8"?>
<Document>
  <Content>
    ...
  </Content>
  <ds:Signature>
    <ds:SignedInfo>
      <ds:Reference URI="">
        <ds:Transforms>
          <ds:Transform
            Algorithm="http://www.w3.org/2000/09/xmldsig
              #enveloped-signature"/>
        </ds:Transforms>
      </ds:Reference>
    </ds:SignedInfo>
    ...
  </ds:Signature>
</Document>
```

XML Encryption

- ▶ Cel: zagwarantowanie poufności danych w XML.
- ▶ Szyfrować można zarówno cały plik XML jak i jego części.

```
<purchaseOrder>
  <Order>
    <Item>book</Item>
    <Id>123-958-74598</Id>
    <Quantity>12</Quantity>
  </Order>
  <Payment>
    <CardId>123654-8988889-9996874</CardId>
    <CardName>visa</CardName>
    <ValidDate>12-10-2004</ValidDate>
  </Payment>
</purchaseOrder>
```

XML Encryption – przykład 1 (fragment)

```
<PurchaseOrder>
  <Order>
    <Item>book</Item>
    <Id>123-958-74598</Id>
    <Quantity>12</Quantity>
  </Order>
  <Payment>
    <CardId>
      <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Content'
        xmlns='http://www.w3.org/2001/04/xmlenc#' >
        <CipherData>
          <CipherValue>A23B45C564587</CipherValue>
        </CipherData>
      </EncryptedData>
    </CardId>
    <CardName>visa</CardName>
    <ValidDate>12-10-2004</ValidDate>
  </Payment>
</PurchaseOrder>
```

XML Signature – przykład 2 (cały dokument)

```
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
  Type='http://www.isi.edu/in-notes/iana/
    assignments/media-types/text/xml'>

  <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'>
    <ds:KeyName>John Smith</ds:KeyName>
  </ds:KeyInfo>

  <CipherData>
    <CipherValue>A23B45C56...56F47345</CipherValue>
  </CipherData>
</EncryptedData>
```