Web Services Technologies: XML, SOAP and UDDI

European University Viadrina
.NET and Web Services Seminar
Adam Twardoch
November 12, 2002
Contents

1. Introduction
2. XML
3. SOAP
4. WDSL
5. UDDI
6. Conclusion
1. Introduction

- Web Services
- Core Technologies
- Pre-Existing Technologies
Web Services

- **Traditional scenario in WWW:**
  Person sitting on a client computer invokes a Web browser and connects to a Web server using HTTP. The Web server returns a page in HTML. Manual interaction.

- **New scenario in WWW (Web services):**
  A client application connects to a Web server using HTTP and invokes remote operations on the Web server. The Web server returns information in XML. Automatic interaction.
Core Technologies

- **XML** (eXtensible Markup Language) to describe all information
- **WDSL** (Web Services Description Language, “wizdel”) to specify Web service interfaces (XML-based)
- **SOAP** (Simple Object Access Protocol) to transport data (XML-based)
- **UDDI** (Universal Description, Discovery, and Integration) to locate information about available Web services, their publishers, interfaces etc. (database with SOAP-based interface)
Core Technologies

Source: Chappell
Core Technologies

- Web services are based on published standards
- XML was created by a broad initiative of vendors and institutions and is now a Web standard owned and published by W3C (the World Wide Web Consortium)
- WDSL, SOAP and UDDI were developed by Microsoft, IBM and several smaller vendors, but were filed for standardization with W3C
- SOAP is now owned by W3C
- Web services are endorsed by various vendors: Microsoft, IBM, Oracle, Sun et al.
- .NET is one possible implementation of Web services
Pre-Existing Technologies

- **(D)COM** (Distributed Component Object Model): architecture that Windows applications use to work together within the same computer (COM) or over networks (DCOM), developed by Microsoft; available on Windows
- **CORBA** (Common Object Request Broker Architecture): open, vendor-independent architecture that applications use to work together over networks, developed by OMG; available mostly on Unix systems
- **Java RMI** (Remote Method Invocation): architecture for Java applications to work together over networks, developed by JavaSoft; only available in Java
2. XML

- History
- Core Concepts
- Structure
- Applications
XML

- Web standard owned and published by W3C (the World Wide Web Consortium)
- All Web services technologies use XML
- Provides a way to describe structured information (data) that can be exchanged between heterogeneous systems
XML: History

- XML is 25 years old
  - In 1969, Charles F. Goldfarb, Ed Mosher, Raymond Morie and Ted Peterson of the IBM Scientific Center in Cambridge, MA invented GML (Generalized Markup Language) to solve problems with describing the structure of legal documents; published in 1973
  - In 1978, ANSI (the American National Standards Institute) established the Computer Languages for the Processing of Text committee
  - In 1980, the first working draft of the SGML (Standard Generalized Markup Language) standard was published
  - In 1986, SGML was published as international standard ISO 8879:1986
  - In 1989, Tim Berners-Lee develops the presentation-oriented Hypertext Markup Language (HTML) as an application of SGML
XML: History

- XML is 25 years old
  - In 1994, W3C (the World Wide Web Consortium) is established
  - In 1995, Yuri Rubinsky and Jon Boask take on development of XML (eXtensible Markup Language) as a subset of SGML
  - In 1998, W3C publishes XML as recommendation
  - In 1998, W3C publishes SMIL (Synchronized Multimedia Integration Language) as first XML-based language
  - In 2000, W3C publishes XHTML that is HTML rewritten in XML
  - Since 2001, numerous XML-based languages such as SVG (Scalable Vector Graphics), MathML (Mathematic Markup Language) et al. are published
XML: Core Concepts

- XML is a markup language for structured documentation (Walsh).

- **Structured documents** are documents that contain both content (words, pictures, animations, drawings, mathematical data etc.) and some indication of what role that content plays. Almost all documents have some structure.

- An **XML document** contains one or more **elements**, each of which is demarcated using **tags**.

- XML documents are not only “text documents”: any kind of digital data can be expressed using XML.
XML: Core Concepts

- “Basically, XML is a subset of SGML that Microsoft engineers could understand”
- XML is **not** a replacement for HTML
- SGML and XML provide a general framework that can be used to describe structured information of any kind
- SGML and XML can be used to build special-purpose languages to describe structured information of a particular kind
- HTML is one such language, along with SVG, MathML, WDSL, WML etc.
XML: Core Concepts

- XML does not DO anything
- XML was not designed to DO anything

Example:

<note>
  <to>Michael</to>
  <from>Adam</from>
  <heading>Confirmation</heading>
  <body>I have received your resume.</body>
</note>
<order>
  <date>2002/11/12</date>
  <customer>Adam Twardoch</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300" />
</order>
XML: Structure

```
<order>
  <date>2002/11/12</date>
  <customer>Adam Twardoch</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300"/>
</order>
```
The text between the opening and closing tags of an element is content.
XML: Structure

Attribute
The start tag for an element can also contain one or more attributes. Each attribute describes some aspect of the element in which it appears.

<order>
  <date>2002/11/12</date>
  <customer>Adam Twardoch</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300" />
</order>
XML: Structure

<order>
  <date>2002/11/12</date>
  <customer>Adam Twardoch</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300" />
</order>

Empty elements
To indicate elements with no content, an optional contraction is allowed. Instead of the closing tag, a slash at the end of the opening tag can be used. Here, it is equivalent to:
<due sum="300"/>"
<order>
  <date>2002/11/12</date>
  <customer>Adam Twardoch</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300" />
</order>
<order>
  <!-- Date will be included automatically -->
  <customer>&author;</customer>
  <items>
    <item no="1" price="100" qty="1">Table</item>
    <item no="2" price="25" qty="4">Chair</item>
    <item no="3" price="100" qty="1">Shelf</item>
  </items>
  <due sum="300" />
</order>

Comments
XML documents can contain comments that are ignored by applications

Entities
The content of the XML document can include entities that are "symbols" referencing other items
XML: Structure Definition

- Element tags must be properly **nested**:
  
  `<name><last> Twardoch </last></name>`

  `<name><last> Twardoch </name></last>`

- XML documents that conform to the general-purpose XML definition rules are called **well-formed**

- The "**grammar**" of special-purpose XML documents are defined using the **DTD** (Document Type Definition) or **XSD** (XML Schema Definition) language

- XML documents that conform to a special-purpose grammar rules are called **valid**
XML: Namespaces

- Provides a **unique identifier for a group of names**
- Used to mix elements from two or more schemas in a single document
- Identified by the `xmlns` attribute:
  ```xml
  <math xmlns="http://www.w3.org/1998/MathML">
    <semantics>
      <mrow>
        ...<mrow>
  </math>
  </semantics>
  ```
- Optionally, all elements from one schema can be **prefixed** with a namespace prefix followed by colon (:)`
  ```xml
  <m:math>
    <m:semantics>
      <m:mrow>
  </m:mrow>
  ```
\[
\sum_{i=0}^{n} x_i
\]
XML Applications: xHTML

\[
\sum_{i=0}^{n} x_i
\]
XML Applications: SVG

\[
\sum_{i=0}^{n} x_i
\]
3. SOAP
SOAP

- **Simple Object Access Protocol**
- At the core of Web services
- XML-based dialect
- uses XML to define a standard format for the data transported using this protocol, i.e. provides a standard way of packaging messages
- SOAP messages mostly sent using HTTP
SOAP: Advantages over COM etc.

- Not tightly coupled to one language
- Not tightly coupled to a particular transport protocol
- Not tied to any one distributed object infrastructure
- Leverages existing industry standards
- Enables interoperability across multiple environments
SOAP: Example

```xml
<?xml version="1.0"?>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <soap:Header>
    <!-- Optional header information goes here. -->
    <To>Scott</To>
    <From>Suzanne</From>
  </soap:Header>

  <soap:Body>
    <!-- Message goes here. -->
    Please pick up some milk on your way home from work.
  </soap:Body>

</soap:Envelope>
```
SOAP Example: Add

```csharp
public int Add(int x, int y)
{
    return x + y;
}
```
SOAP Example: Add Request

```xml
<?xml version="1.0"?>
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
><soap:Body>
    <Add>
        <x>1</x>
        <y>2</y>
    </Add>
</soap:Body>
</soap:Envelope>
```
SOAP Example: Add Response

```xml
<?xml version="1.0"?>
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
><soap:Body>
  <AddResult>
    <result>1</result>
  </AddResult>
</soap:Body>
</soap:Envelope>
```
4. WDSL
WDSL

- **Web Service Description Language** ("wizdel")
- XML-based dialect
- Provides a standardized way to specify features such as the operations in an interface or the input and output parameters of those operations
- A WSDL document provides the information necessary for a client to interact with the Web service
- Can be used to describe practically any network service, including SOAP over HTTP and even protocols that are not XML-based, such as DCOM over UDP.
WDSL: Structure

- A WSDL document is composed of five elements under the root `definitions` element:
  - `types`
  - `message`
  - `portType`
  - `binding`
  - `service`
- These elements are used to define Web services through a series of associations.
- Software tools such as Visual Studio.NET can generate WDSL automatically based on an existing Web service.
<?xml version="1.0" encoding="utf-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <element name="Add"><complexType>
      <all><element name="x" type="int"/><element name="y" type="int"/></all>
    </complexType></element>
    <element name="AddResult"><complexType>
      <all><element name="result" type="int"/></all>
    </complexType></element>
  </types>
  <message name="AddMsgIn"><part name="parameters" element="s:Add"/></message>
  <message name="AddMsgOut"><part name="parameters" element="s:SubtractResult"/></message>
  <portType name="CalculatorPortType">
    <operation name="Add"><input message="tns:AddMsgIn"/><output message="tns:AddMsgOut"/></operation>
  </portType>
  <service name="CalculatorService">
    <port name="CalculatorPort" binding="tns:CalculatorBinding">
      <soap:address location="http://somedomain/Calculator"/></port>
  </service>
</definitions>
5. UDDI
UDDI

- Universal Description, Discovery, and Integration
- Used to locate information about available Web services, their publishers, interfaces etc.
- XML-based **UDDI business registration** documents that are stored in a replicated database known as the **UDDI business registry**
- Business registry is a **Web service**
- **Standard API** to modify entries in the registry
UDDI: Structure

- one **businessEntity** element
- contains one or more **businessService** elements
- these contain one or more **bindingTemplate** elements

```xml
<businessEntity>
  <name>QwickBank</name>
  <description>Internet bank</description>
  <bindingTemplates>
    <bindingTemplate bindingKey="9E43F20A-1963-22EC-1053-444553540000">
      <accessPoint>
        http://www.qwickbank.com/accounts.asmx
      </accessPoint>
    </bindingTemplate>
    ...
  </bindingTemplates>
</businessEntity>
```
UDDI: Problems

- “complex solution to what should be a simple problem”: finding WSDL definitions for Web services all over the Internet
- .NET Framework includes Disco, a much simpler technology for discovering Web services
- Visual Studio.NET has built-in support for both UDDI and Disco
6. Conclusion
Conclusion: Known problems

- SOAP allows adding arbitrary headers, these are non-standard.
- In late 2001, Microsoft announced Global XML Web Services Architecture (GXA) that defines a set of SOAP headers for addressing common problems such as security and routing.
- Alternative solutions to .NET are not mature yet.