A7B36XML, AD7B36XML

XML Technologies

Lecture 2

Namespaces, InfoSet, XML Formats

10. 3. 2017

Authors: **Irena Holubová, Marek Polák**
Lecturer: **Martin Svoboda**

http://www.ksi.mff.cuni.cz/~svoboda/courses/2016-2-A7B36XML/

Lecture Outline

- XML
  - Namespaces
  - Data models
- **Standard XML formats**

Namespaces

Namespaces

Today we learn what it is, later (when talking about XML Schema) we learn how to create and use it.

- Problem: We need to distinguish the same names of elements and attributes in cases when a conflict may occur.
  - The application needs to know which elements/attributes it should process
  - e.g. name of a book vs. name of a company
- Idea: expanded name of an element/attribute = ID of a namespace + local name
  - The namespace is identified by URI
- URI is too long → shorter version
  - Namespace declaration = prefix + URI
  - Qualified name (QName) = prefix + local name of an element/attribute
- Note: DTD does not support namespaces (it considers them as any other element/attribute names)
  - XML Schema is conversely based on namespaces

Ex. Namespace

Namespace declaration → the area of validity

```
<pricelist:offer
    xmlns:pricelist="http://www.eprice.cz/e-pricelist">
  <pricelist:item tax="22%">
    <pricelist:name>
      <bib:book xmlns:bib="http://www.my.org/bib">
        <bib:author>Mark Logue</bib:author>
        <bib:name>The King's Speech: How One Man Saved
the British Monarchy</bib:name>
      </bib:book>
    </pricelist:name>
    <pricelist:price curr="CZK">259</pricelist:price>
  </pricelist:item>
</pricelist:offer>
```

Namespace usage

Ex. Implicit Namespace

```
<offer
    xmlns="http://www.eprice.cz/e-pricelist">
  <item tax="22%">
    <name>
      <bib:book xmlns:bib="http://www.my.org/bib">
        <bib:author>Mark Logue</bib:author>
        <bib:name>The King's Speech: How One Man
Saved the British Monarchy </bib:name>
      </bib:book>
    </name>
    <price curr="CZK">259</price>
  </item>
</offer>
```

Namespace

- A set of non-conflicting identifiers
- A namespace consists of disjoint subsets:
  - All element partition
    - A unique name is given by namespace identifier and element name
    - I.e. all elements have unique names
  - Per element type partitions
    - A unique name is given by namespace identifier, element name and local name of attribute
    - I.e. attributes have names unique within element declarations
  - Global attribute partition
    - A unique name is given by namespace identifier and attribute name
      - This kind of attribute can be defined in XML Schema
    - I.e. a special type of attributes having unique names among all attributes

See XML Schema for explanation

Ex. Parts of Namespaces

```
<offer
    xmlns="http://www.eprice.cz/e-pricelist"
    xmlns:bib="http://www.my.org/bib">
  <item tax="22%">
    <name>
      <bib:book>
        <bib:author>Mark Logue</bib:author>
        <bib:name xml:lang="cs">The King's Speech:
How One Man Saved the British  archy </bib:name>
      </bib:book>
    </name>
    <price curr="CZK">259</price>
  </item>
</offer>
```

Element from namespace bib

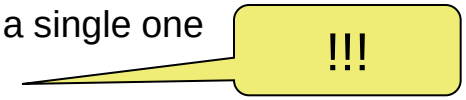
???

Global attribute from namespace xml

Attribute of element price, from (implicit) namespace http://www.eprice.cz/e-pricelist

Namespace XML

- Each XML document is assigned with namespace XML
  - URI: http://www.w3.org/XML/1998/namespace
  - Prefix: xml
  - It does not have to be declared
- It involves global attributes:
  - xml:lang – the language of element content
    - Values are given by the XML specification
  - xml:space – processing of white spaces by the application
    - preserve
    - default = use application settings
      - Usually replaces multiple white spaces with a single one
  - xml:id – unique identifier (of type ID)
  - xml:base – declaration of base URI, others can be defined relatively
    - E.g. in XML technology XLink



More on Namespaces

- W3C specification:
  - http://www.w3.org/TR/REC-xml-names/
- Lectures on XML Schema
  - Later

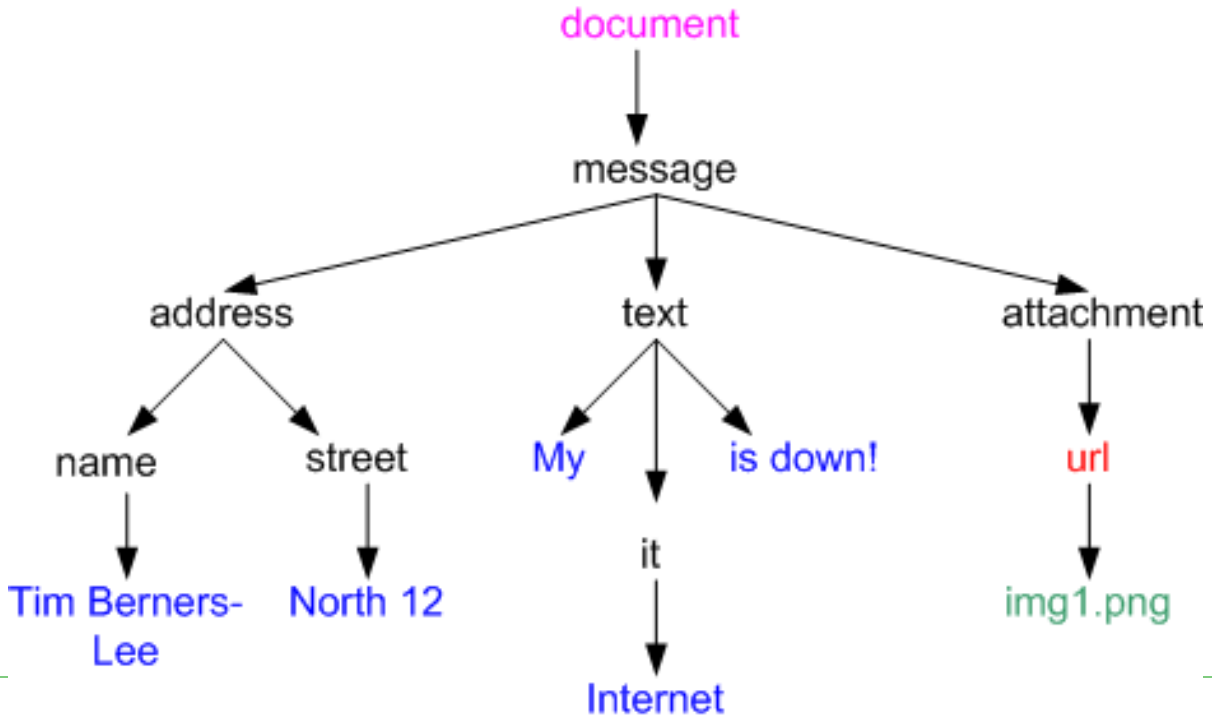
XML Data Model

XML Infoset

- A well formed XML document → hierarchical tree structure = XML Infoset
  - Abstract data model of XML data
- Information set = the set of information (in the XML document)
- Information item = a node of the XML tree
  - Types of items: document, element, attribute, string, processing instruction, comment, notation, DTD declaration, …
  - Properties of items: name, parent, children, content, …
- It is used in other XML technologies
- DTD (in general XML schema) can „modify“ Infoset
  - E.g. default attribute values

Ex.

```
<message>
  <address>
    <name>Tim Berners-Lee</name>
    <street>North 12</street>
  </address>
  <text>My <it>Internet</it> is down!</text>
  <attachment url="img1.png"/>
</message>
```



Ex. Element Information Item

- [namespace name]
  - (Possibly empty) name of namespace
- [local name]
  - Local part of element name
- [prefix]
  - (Possibly empty) prefix of namespace
- [children]
  - (Possibly empty) sorted list of child items
    - Document order
    - Elements, processing instructions, unexpanded references to entities, strings and comments
- [attributes]
  - (Possibly empty) unsorted set of attributes (Attribute Information Items)
    - Namespace declarations are not included here
  - Each item (attribute) is declared or given by the XML schema
    - Attributes with default values

Ex. Element Information Item

- [namespace attributes]
  - (Possibly empty) unsorted set of declarations of namespaces
- [in-scope namespaces]
  - Unsorted set of namespaces which are valid for the element
  - It always contains namespace XML
  - It always contains items of set [namespace attributes]
- [base URI]
  - URI of the element
- [parent]
  - Document/Element Information Item to whose property [children] the element belongs

- For other items see http://www.w3.org/TR/xml-infoset/

Post Schema Validation Infoset (PSVI)

- Typed Infoset
- It results from assigning data types on the basis of validation against an XML schema
  - We can work directly with typed values
  - Without PSVI we have only text values
    - DTD: minimum of data types
    - XML Schema: int, long, byte, date, time, boolean, positiveInteger, …
- Usage: in query languages (XQuery, XPath)
  - E.g. We have functions specific for strings, numbers, dates etc.

XML Formats

Standard XML Formats

- XML schema = a description of allowed structure of XML data = XML format
  - DTD, XML Schema, Schematron, RELAX NG, …
- Standard XML format = a particular XML schema which became standard for a particular (set of) application(s)
  - Input/output data must conform to the format
  - Usually it is an acknowledged standard

Standard XML Formats

- Categorization:
  - Publication of data on the Web
    - XHTML, MathML, SVG, XForms
  - Office SW
    - Office Open, OpenDocument
  - Technical documentation
    - DocBook
  - Data exchange in communities
    - UBL, OpenTravel
  - Web services
    - SOAP, WSDL, UDDI
  - And many other…

Publication of data on the Web

eXtensible HyperText Markup Language (XHTML)

- http://www.w3.org/TR/xhtml1/
- Results from HTML
  - XHTML 1.0 corresponds HTML 4.01
    - Adapts HTML so that it corresponds to XML standard
      - Well-formedness
- XHTML document must:
  1. Be valid against one of three XHTML DTDs which are explicitly referenced using DOCTYPE declaration
  2. Have root element html
  3. Declare namespace of HTML: http://www.w3.org/1999/xhtml

Sample XHTML Document

Moved to example.org.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
       xml:lang="en" lang="en">
  <head>
    <title>Virtual Library</title>
  </head>
  <body>
    <p>Moved to <a
href="http://example.org/">example.org</a>.</p>
  </body>
</html>
```

XHTML vs. HTML

- The documents must be well-formed
  - End tags are required
  - Empty elements must have the end tag or the abbreviated version
  - Element tags must be properly nested
  - Attributes must have a value and it must be in quotations
- Names of elements and attributes must be in lower case
- Elements script and style have the #PCDATA type
  - Usage of special characters must correspond to XML rules

XHTML vs. HTML

```
<p>here is an emphasized <em>paragraph</em>.</p>
```

```
<p>here is an emphasized <em>paragraph.</p></em>
```

```
<p>here is a paragraph.</p><p>here is another paragraph.</p>
```

```
<p>here is a paragraph.<p>here is another paragraph.
```

```
<td rowspan="3">
```

```
<td rowspan=3>
```

```
<dl compact="compact">
```

```
<dl compact>
```

```
<br/><hr/>
```

```
<br><hr>
```

```
<script type="text/javascript">
<![CDATA[
... unescaped script content ...
]]>
</script>
```

XHTML DTD

- XHTML 1.0 Strict
  - „Purely“ structural tags of HTML 4.01 + cascading style sheets
- XHTML 1.0 Transitional
  - Constructs from older HTML versions, including those for presentation
    - font, b, i, …
- XHTML 1.0 Frameset
  - Exploitation of frames

XHTML DTD

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

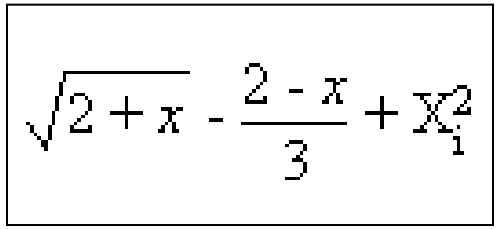
```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

MathML – Mathematical Markup Language

- http://www.w3.org/Math/
- Mathematical equations in XML
- Can be combined with XHTML
  - Browsers mostly support it
- We can transform TeX ↔ MathML
- Elements:
  - Presentation – describe the structure of the equation
    - Upper index, lower index, …
  - Content – describe mathematical objects
    - Plus, vector, ..
  - Interface – combination with HTML, XML, …
- Support: FireFox, Opera

Construct	Meaning
mtext	text
mspace	space
mi	IDs – variables
mn	numbers
mo	operators (+, -, /, *) and parentheses
mtable	table
mrow	row
mtd	column
mfrac	fraction
msqrt	square root
mroot	general root
msub	lower index
msup	upper index
msubsup	lower and upper index

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE math PUBLIC "-//W3C//DTD MathML 2.0//EN"
          "http://www.w3.org/Math/DTD/mathml2/mathml2.dtd">
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <msqrt>
    <mn>2</mn>
    <mo>+</mo>
    <mi>x</mi>
  </msqrt>
  <mo>-</mo>
  <mfrac>
    <mrow><mn>2<mo>-</mo><mi>x</mi></mn></mrow>
    <mrow><mn>3</mn></mrow>
  </mfrac>
  <mo>+</mo>
  <msubsup>
    <mn>X</mn>
    <mrow>
      <mn>i</mn>
    </mrow>
    <mrow>
      <mn>2</mn>
    </mrow>
  </msubsup>
</math>
```



SVG – Scalable Vector Graphics

- http://www.w3.org/Graphics/SVG/
- 2-dimensional vector graphics
- Types of graphical objects:
  - Vector – 2D images, splines
  - Raster images
  - Text objects
- Grouping, formatting, transformations, animation, filtering, …
- Support: FireFox, Opera, MS IE (plug-in)
- Tools: inkscape (e.g.)

Construct	Meaning
rect	rectangle
circle	
ellipse	
line	
polygon	
polyline	
path	spline
text	
font	
animateMotion, animateColor, animateTransform, …	animations
feBlend, feColorMatrix, feDistantLight, …	filters

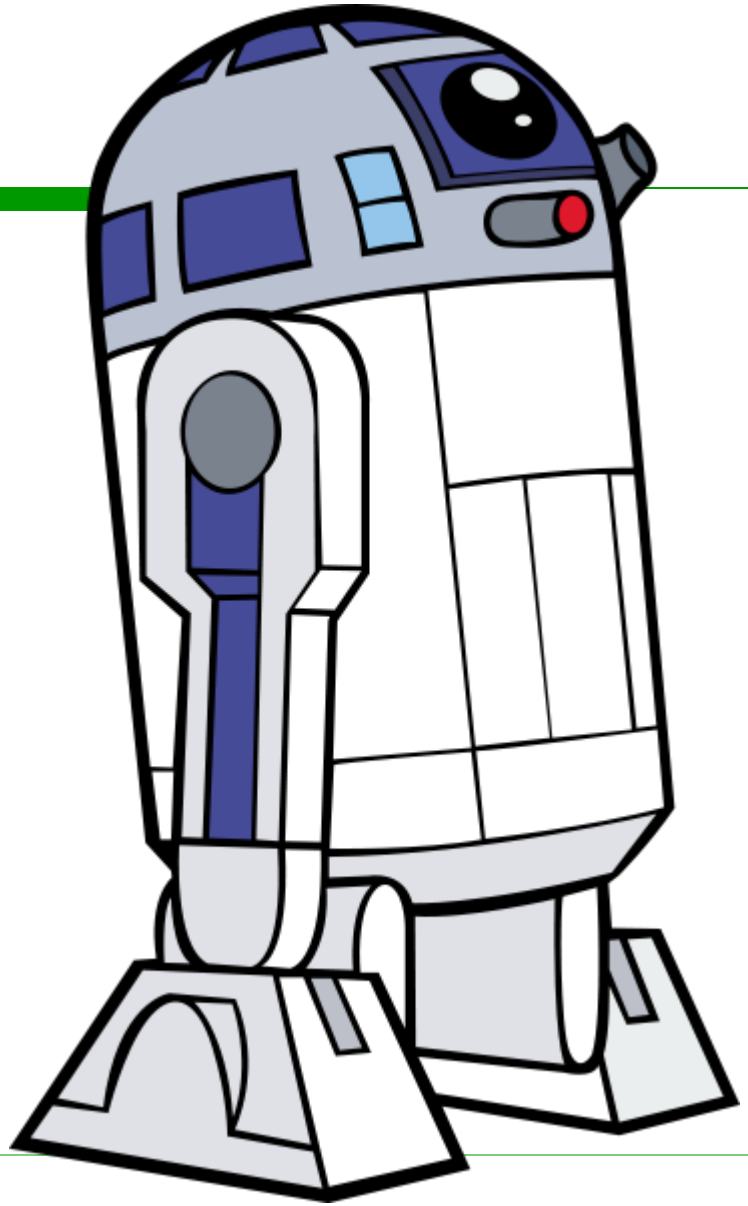
SVG – Example

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
          "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
     width="467" height="462">
  <rect x="80" y="60" width="250" height="250" rx="20"
        style="fill:#ff0000;stroke:#000000;stroke-width:2px;"/>

  <rect x="140" y="120" width="250" height="250" rx="40"
        style="fill:#0000ff;stroke:#000000;stroke-width:2px;
        fill-opacity:0.7;"/>
</svg>
```

SVG – Example



CloneWars.svg

X3D – eXtensible 3D

- http://www.web3d.org/x3d/
  - Web3D Consortium
- http://x3dgraphics.com/examples/index.php
- 3-dimentional vector graphics
  - Visual effects, modelling of behaviour, user interaction
- Successor of VRML (Virtual Reality Modelling Language)
- Support: SwirlX3D Viewer (e.g.)

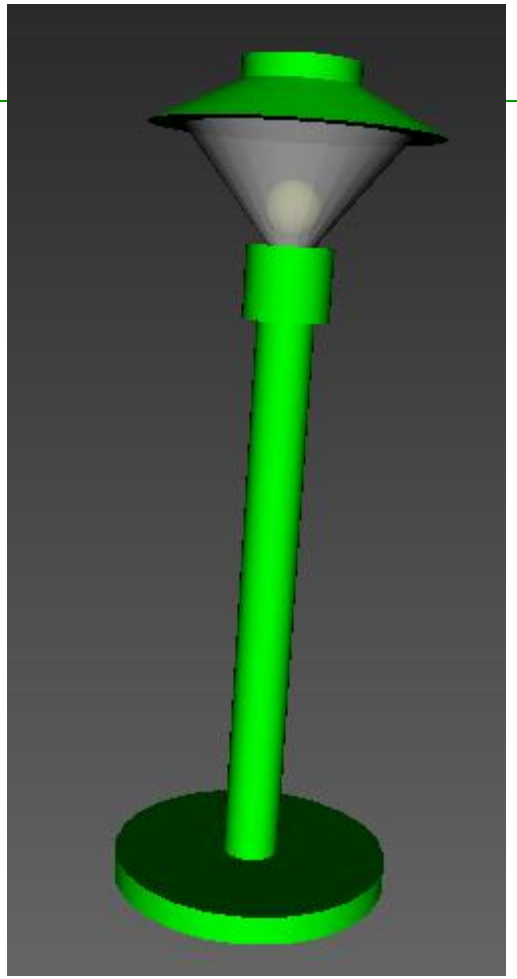
Construct	Meaning
Box	
Cone	
Cylinder	
Sphere	
Text	
Background	
Vievpoint	
Appearance	color, texture, transparency, …
IndexedFaceSet	set of facets
ElevationGrid	

X3D – Example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN"
          "http://www.web3d.org/specifications/x3d-3.0.dtd">

<X3D profile="Immersive" version="2.0">
  <Scene>
    <Transform>
      <Shape>
        <Appearance>
          <Material diffuseColor="0 1 0"/>
        </Appearance>
        <Cylinder height="0.1" radius="0.5"/>
      </Shape>
    </Transform>
  </Scene>
</X3D>
```

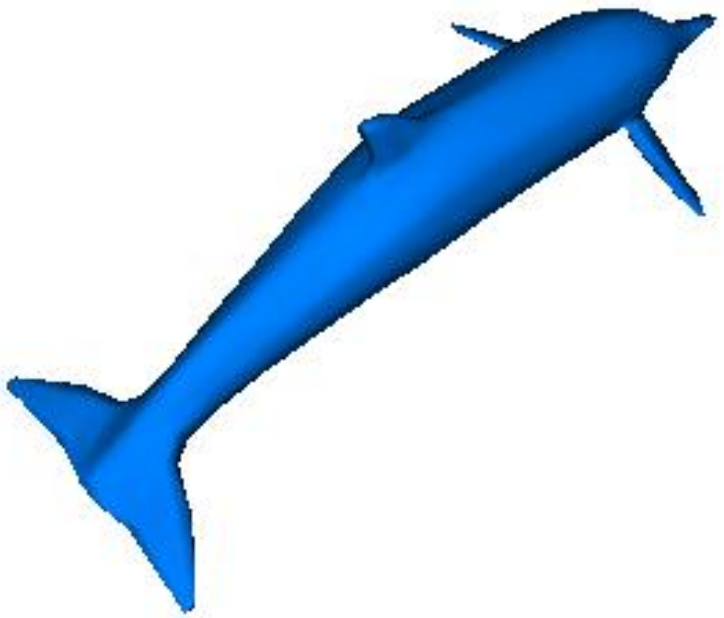
X3D – Example



lamp.x3d

X3D – Example

dolphin.x3d



XForms

- http://www.w3.org/TR/xforms11/
- Description of user interface for XML data – Web forms
- New generation of HTML forms
- XForms Controls
  - Which items of interface should be used
  - Visualization is determined by a particular browser
- Parts:
  - Data model – what data are processed
  - User interface – input/output controls and their features

Construct	Meaning
model	model of input data
instance	description of input data
submission	type of input data
input	single-line input text box
label	label
secret	single-line input edit box with hidden characters
textarea	multi-line input edit box
trigger	running of an action
upload	file upload
output	data output
submit	confirmation button
select	choice from items

XForms – Example

```
...
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:xf="http://www.w3.org/2002/xforms">
  <head>
    <title>Hello World in XForms</title>
    <xf:model>
      <xf:instance>
        <f:data xmlns:f="http://foo.com">
          <f:PersonGivenName/>
        </f:data>
      </xf:instance>
    </xf:model>
  </head>
  ...
```

http://www.agencexml.com/xsltforms/hello.xml

XForms – Example

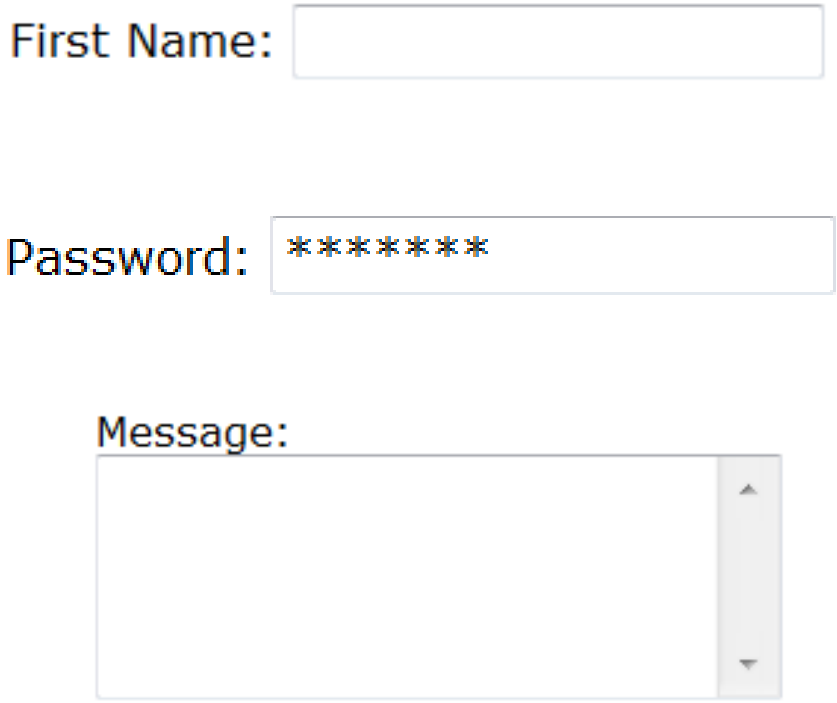
```
  ...
  <body>
    <p>Type your first name in the input box. If you are running XForms,
       the output should be displayed in the output area.</p>
    <p>
      <xf:input ref="f:PersonGivenName" incremental="true">
        <xf:label>Please enter your first name: </xf:label>
      </xf:input>
    </p>
    <p>
      <xf:output value="concat('Hello ',
                               f:PersonGivenName,
                               '. We hope you like XForms!')">
        <xf:label>Output: </xf:label>
      </xf:output>
    </p>
    <div id="console" style="display: block"/>
  </body>
</html>
```

XForms – Further Examples

```
<input ref="name/fname">
  <label>First Name</label>
</input>
```

```
<secret ref="name/password">
  <label>Password:</label>
</secret>
```

```
<textarea ref="message">
  <label>Message</label>
</textarea>
```



Creation of Technical Documentation

DocBook

- http://www.docbook.org/
  - Documentation
- http://www.oasis-open.org/docbook/
  - Current information
- http://docbook.sourceforge.net/
  - Styles
- http://www.kosek.cz/xml/db/
  - Tutorial in Czech
- A format originally designed for SW documentation
- Currently used for writing books, articles, …
- The text is divided into sections, subsections, …
- The language supports constructs for names of programs, files, listings, images, shortcuts, screen shots, …

Construct	Meaning
book	
article	
qandaset	FAQ
refentry	referential pages
bookinfo	information about book
ToC	contents
index	
preface	
chapter	
appendix	
itemizedlist	list of items
programlisting	

Construct	Meaning
equation	
example	
figure	
footnote	
anchor	anchor for a particular position in text
link	reference to an anchor
citation	
emphasis	
table	
thead	head of table
tbody	body of table
row	row of table

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.5//EN'
          'http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd'>
<book lang="cs">
  <bookinfo>
    <title>My First Book</title>
    <author>
      <firstname>Jiri</firstname>
      <surname>Kosek</surname>
    </author>
  </bookinfo>
  <preface>
    <title>Introduction</title>
    <para>Paragraph of text.</para>
    <para>...</para>
  </preface>
  <chapter>
    <title>First Chapter</title>
    <para>Text of first chapter</para>
    <para>...</para>
  </chapter>
  <appendix>
    <title>First Appendix</title>
    <para>Text of first appendix</para>
    <para>...</para>
  </appendix>
</book>
```

Formatting of DocBook Documents

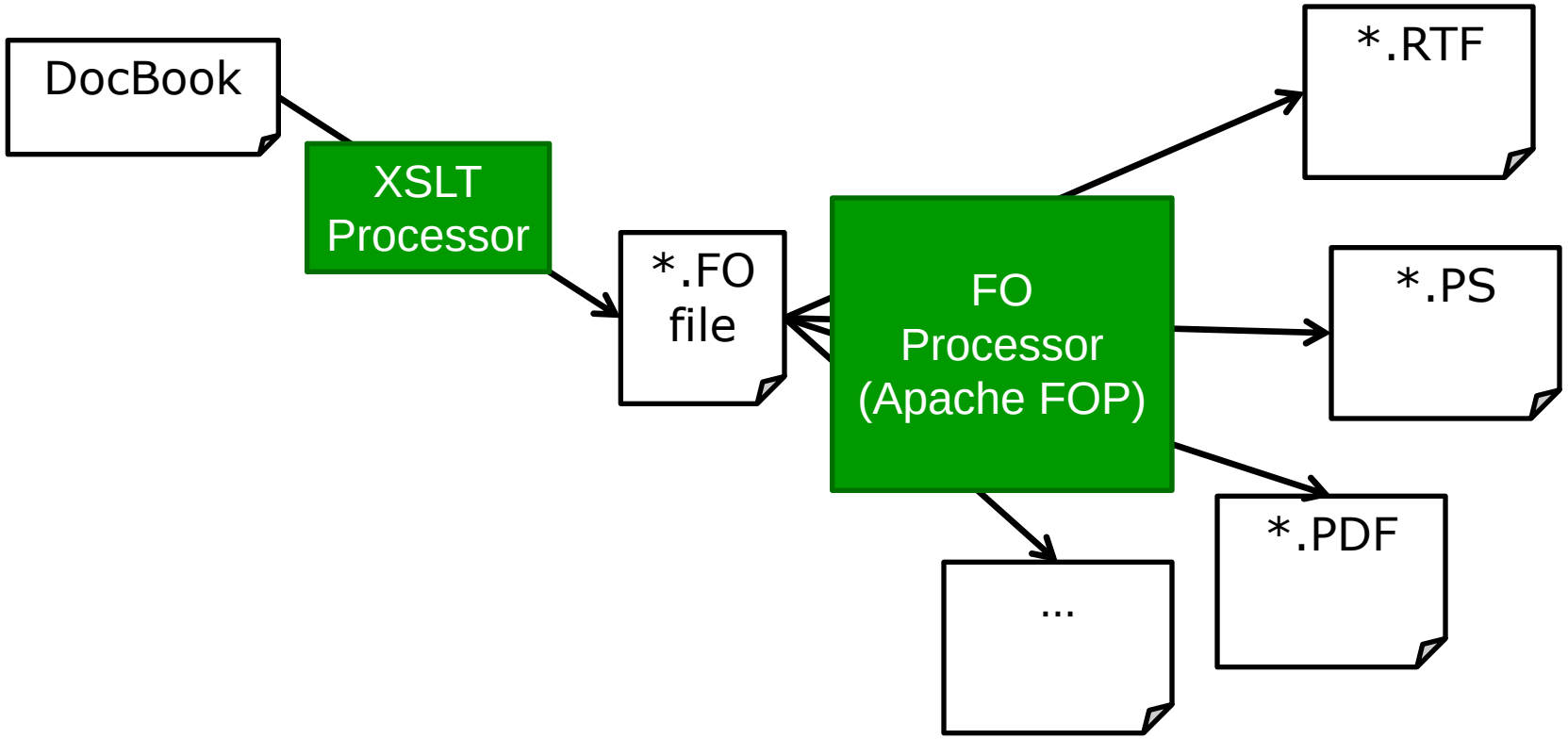
- We need to visualize the constructs
  - We need to specify the font of title, the size of space at the beginning of a paragraph, …
- Options:
  - CSS – too simple, often insufficient
  - XSLT – originally designed for visualization of XML data (but today it is used for more general purposes)
    - There exist free XSL style sheets for DocBook visualization
      - E.g. transformation to XHTML, PDF, …
      - Apache FOP

Support of DocBook in Editors

- XMLmind
  - http://www.xmlmind.com/xmleditor/
  - WYSIWYG editor
- oXygen
  - http://www.oxygenxml.com/
- jEdit
  - http://www.jedit.org/
- Visual Studio, IntelliJ Idea, etc.

Apache FOP (Formatting Objects Processor)

- XSL based tool written in Java



Office SW

Office SW

- DocBook is for technical, expert texts
  - Assumes technically skilled user
  - Not all editors are WYSIWYG
- Standard user: MS Office, OpenOffice, …
- Office Open XML (OOXML)
  - http://openxmldeveloper.org/
- OpenDocument (ODF)
  - http://www.odfalliance.org/
- XPS (XML Paper Specification) – XML based page description language and a fixed-document format

OOXML (Office Open XML)

- Microsoft
  - For the first time in Office 2007
- Parts of specification:
  - Description of structure of files
    - In general a zip file with XML + other data
  - XML for text editors (file extension docx)
  - XML for spreadsheets (xlsx)
  - XML for presentations (pptx)
  - XML for graphics – DrawingML
  - XML for special items – MS MathML
    - e.g. equations

why not SVG?

why not W3C MathML?

OOXML – Example

This is plain text in the header. The next line contains a centered "AutoText" field.
Office Open XML sample.doc

Normal

Heading 1

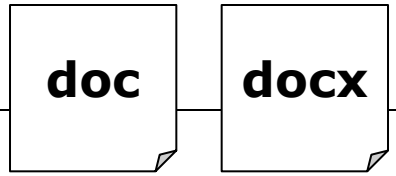
Heading 2

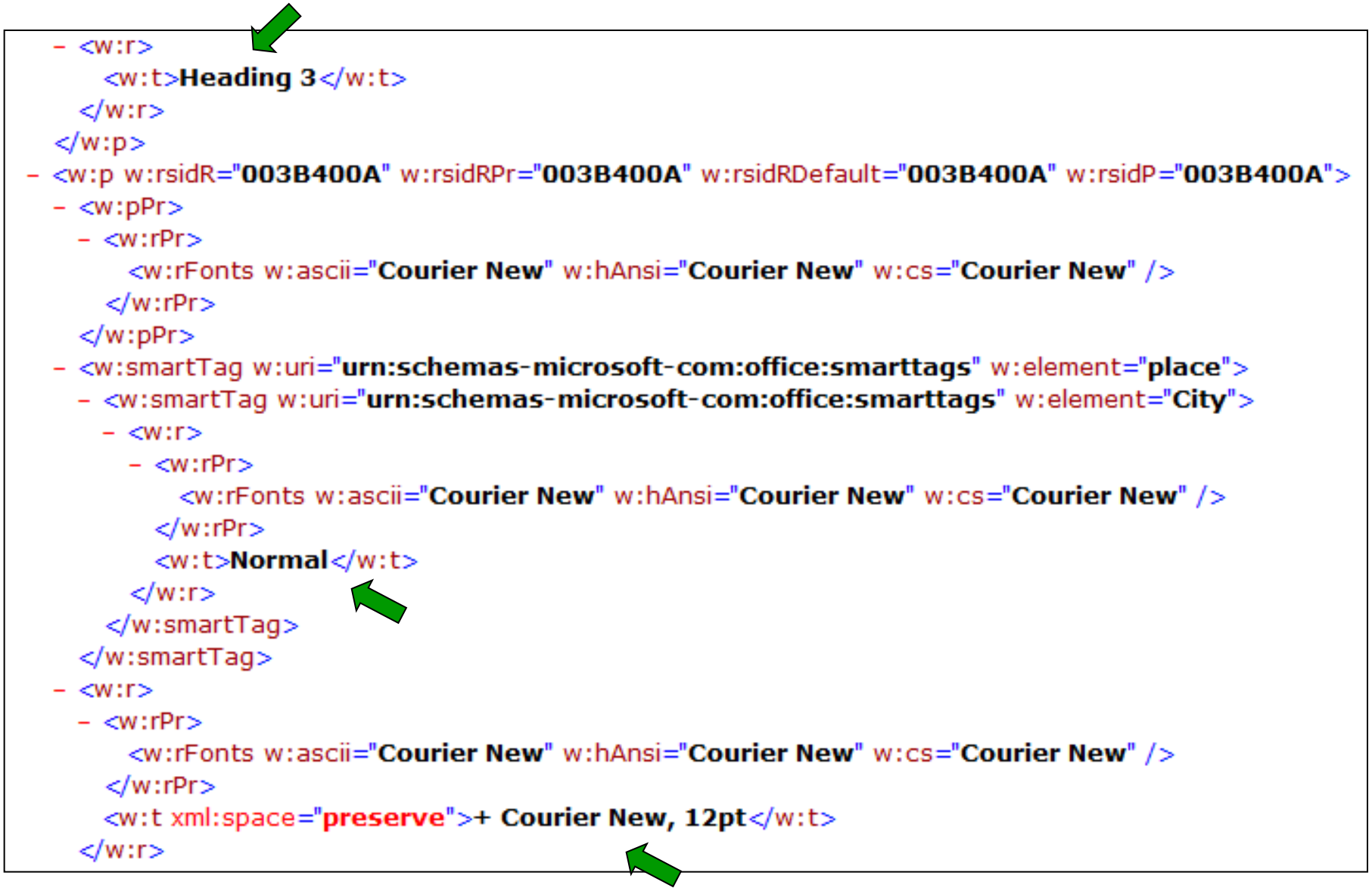
Heading 3

Normal + Courier New, 12pt
Normal + Courier New, 12pt, Bold
Normal + Courier New, 12pt, Italic
Normal + Courier New, 12pt, Underlined
Normal + Courier New, 12pt, Bold + Italic
Normal + Courier New, 12pt, Bold + Underlined
Normal + Courier New, 12pt, Italic + Underlined
Normal + Courier New, 12pt, Bold + Italic + Underlined

Arial, 8pt
Arial, 9pt
Arial, 10pt
Arial, 11pt
Arial, 12pt

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <w:wordDocument xmlns:ve="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:o="urn:schemas-microsoft-com:office:office"
    xmlns:o12="http://schemas.microsoft.com/office/2004/7/core"
    xmlns:r="http://schemas.openxmlformats.org/officeDocument/2006/relationships"
    xmlns:m="http://schemas.microsoft.com/office/omml/2004/12/core" xmlns:v="urn:schemas-microsoft-com:vml"
    xmlns:wp="http://schemas.openxmlformats.org/drawingml/2006/3/wordprocessingDrawing"
    xmlns:w10="urn:schemas-microsoft-com:office:word"
    xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/2/main">
    <w:smartTagType w:namespaceuri="urn:schemas-microsoft-com:office:smarttags" w:name="place" />
    <w:smartTagType w:namespaceuri="urn:schemas-microsoft-com:office:smarttags" w:name="country-region" />
    <w:smartTagType w:namespaceuri="urn:schemas-microsoft-com:office:smarttags" w:name="City" />
  - <w:body>
    - <w:p w:rsidR="003B400A" w:rsidRDefault="003B400A">
      - <w:smartTag w:uri="urn:schemas-microsoft-com:office:smarttags" w:element="place">
        - <w:smartTag w:uri="urn:schemas-microsoft-com:office:smarttags" w:element="City">
          - <w:r>
              <w:t>Normal</w:t>
            </w:r>
          </w:smartTag>
        </w:smartTag>
      </w:p>
    - <w:p w:rsidR="003B400A" w:rsidRDefault="003B400A" w:rsidP="003B400A">
      - <w:pPr>
          <w:pStyle w:val="Heading1" />
        </w:pPr>
      - <w:r>
          <w:t>Heading 1</w:t>
        </w:r>
      </w:p>
```




- <w:r>
<w:t>Heading 3</w:t>
</w:r>
</w:p>
- <w:p w:rsidR="003B400A" w:rsidRPr="003B400A" w:rsidRDefault="003B400A" w:rsidP="003B400A">
- <w:pPr>
- <w:rPr>
<w:rFonts w:ascii="Courier New" w:hAnsi="Courier New" w:cs="Courier New" />
</w:rPr>
</w:pPr>
- <w:smartTag w:uri="urn:schemas-microsoft-com:office:smarttags" w:element="place">
- <w:smartTag w:uri="urn:schemas-microsoft-com:office:smarttags" w:element="City">
- <w:r>
- <w:rPr>
<w:rFonts w:ascii="Courier New" w:hAnsi="Courier New" w:cs="Courier New" />
</w:rPr>
<w:t>Normal</w:t>
</w:r>
</w:smartTag>
</w:smartTag>
- <w:r>
- <w:rPr>
<w:rFonts w:ascii="Courier New" w:hAnsi="Courier New" w:cs="Courier New" />
</w:rPr>
<w:t xml:space="preserve">+ Courier New, 12pt</w:t>
</w:r>

ODF

- OASIS
- ISO standard
  - From the beginning an open format
  - OpenOffice, StarOffice, EuroOffice, MS Office, Corel WordPerfect Office Suite, KOffice, TextEdit, …
- File extensions:
  - odt = text
  - ods = spreadsheet
  - odp = presentation
  - odc = graph
  - odi = figure
  - …

ODF – Example

odt

AOPK ČR
Správa CHKO Broumovsko
Ledhujská 59
549 54 Police nad Metují

ŽÁDOST

o vydání souhlasu k umístění nebo povolení stavby na území CHKO Broumovsko podle zákona č. 114/1992 Sb.

A. Žadatel

(Poučení: Pokud zamýšlíte využít rozhodnutí o souhlasu k umístění nebo povolení stavby na území CHKO Broumovsko jako podklad v řízení před stavebním úřadem /zejména v územním nebo stavebním řízení/, musí být žadatel o vydání souhlasu podle zákona č. 114/1992 Sb. totožný se stavebníkem v řízení podle stavebního zákona.

Žadatel se může nechat v řízení zastoupit na základě písemné plné moci, kterou je třeba k žádosti přiložit. V takovém případě uvádějte v této žádosti vždy identifikační údaje žadatele, nikoliv jeho zástupce; identifikační údaje zástupce musí být uvedeny v plné moci.)

- Jméno a příjmení (název) žadatele: .....

ODF – Example

```
- <text:p text:style-name="P25">
    o vydání souhlasu k umístění nebo povolení stavby na území CHKO Broumovsko podle
    <text:s />
    zákona
    <text:line-break />
    č. 114/1992 Sb.
  </text:p>
  <text:p text:style-name="Standard" />
  <text:h text:style-name="P26" text:outline-level="1" text:is-list-header="true">A. Žadatel</text:h>
  <text:p text:style-name="P40">(Poučení: Pokud zamýšlíte využít rozhodnutí o souhlasu k umístění nebo
    povolení stavby na území CHKO Broumovsko jako podklad v řízení před stavebním úřadem /zejména v
    územním nebo stavebním řízení/, musí být žadatel o vydání souhlasu podle zákona č. 114/1992 Sb.
    totožný se stavebníkem v řízení podle stavebního zákona.</text:p>
  <text:p text:style-name="P40">Žadatel se může nechat v řízení zastoupit na základě písemné plné moci,
    kterou je třeba k žádosti přiložit. V takovém případě uvádějte v této žádosti vždy identifikační údaje
    žadatele, nikoliv jeho zástupce; identifikační údaje zástupce musí být uvedeny v plné moci.)</text:p>
- <text:list text:style-name="RTF_5f_Num_20_11">
  - <text:list-item>
    - <text:p text:style-name="P10">
        <text:span text:style-name="T1">Jméno a příjmení</text:span>
        <text:span text:style-name="T2">(název)</text:span>
      - <text:span text:style-name="T1">
          žadatele:
          <text:tab />
          <text:tab />
          .....
        </text:span>
      </text:p>
```

Formats for Communities

Formats for Communities

- There are many communities which exchange data using XML format
  - Commerce, travelling, banking, health, chemistry, genetic engineering, …
- Beforehand they agree on a particular format of messages
  - Data structures
- Universal Business Language (UBL)
  - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ubl
  - Exchange of business data
- OpenTravel
  - http://www.opentravel.org/
  - Exchange of data in tourism

UBL – Example

```
<po:Order xmlns:po="urn:oasis:names:tc:ubl:Order:1.0:0.70"
          xmlns="urn:oasis:names:tc:ubl:CommonAggregateTypes:1.0:0.70">
  <ID>123456789</ID>
  <IssueDate>2008-02-26</IssueDate>
  <BuyerParty>
    <ID>B001</ID>
    <PartyName>
      <Name>Martin Nečaský</Name>
    </PartyName>
    <Address>
      <Street>Malostranské nám. 25</Street>
      <CityName>Praha</CityName>
      <Country>Czech republic</Country>
    </Address>
  </BuyerParty>
  <OrderLine>
    <Quantity unitCode="unit">1</Quantity>
    <Item>
      <ID>I123</ID>
      <Description>Technologie XML</Description>
      <BasePrice>
        <PriceAmount currencyID="CZK">1</PriceAmount>
      </BasePrice>
    </Item>
  </OrderLine>
</po:Order>
```

OpenTravel – Example

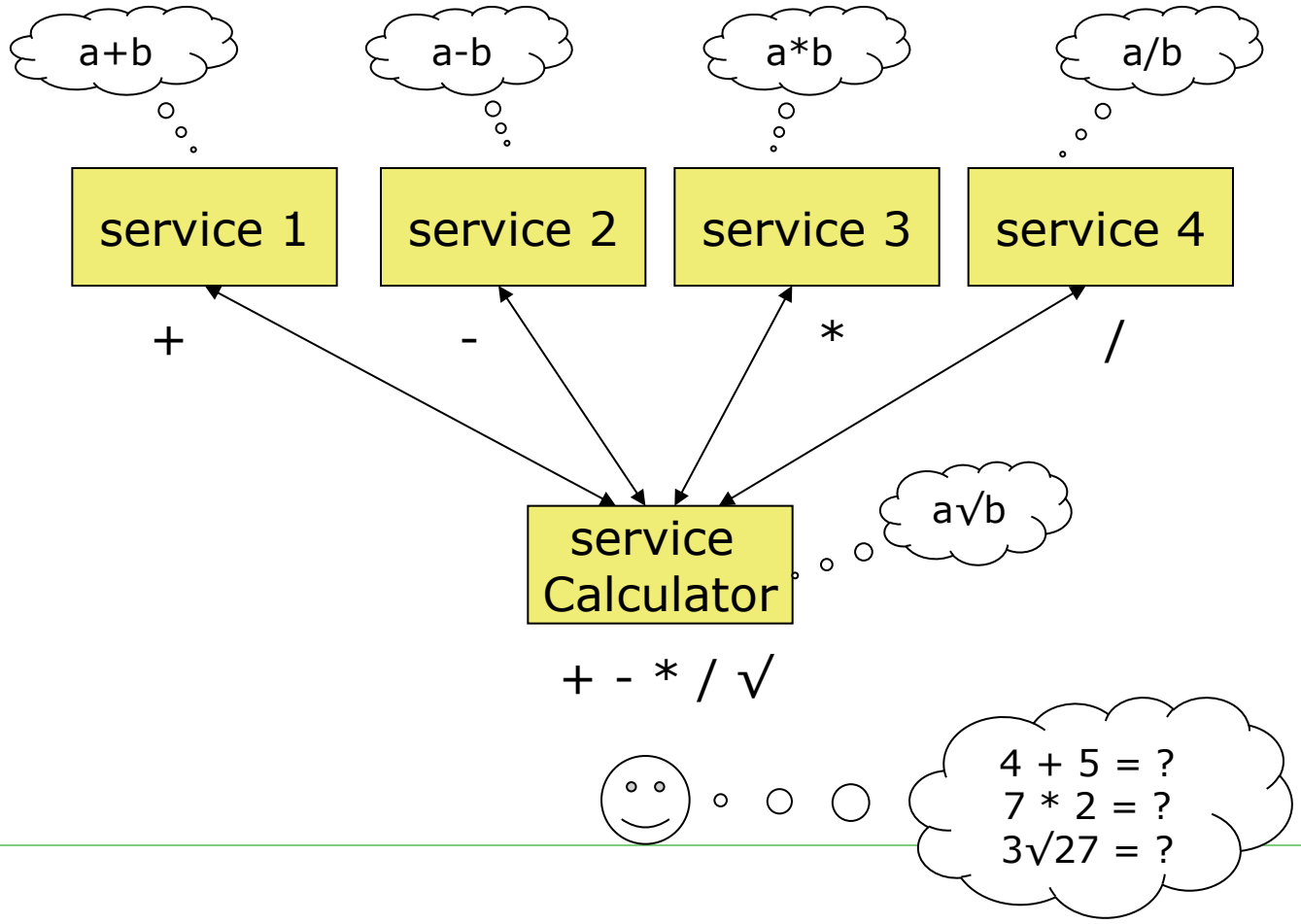
```
<OTA_HotelResRQ xmlns="http://www.opentravel.org/OTA/2003/05"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05_OTA_
                                    HotelResRQ.xsd" Version="1.003">
  <POS>
    <Source ISOCurrency="USD"/>
  </POS>
  <HotelReservations>
    <HotelReservation>
      <RoomStays>
        <RoomStay>
          <RoomTypes>
            <RoomType NumberOfUnits="1"/>
          </RoomTypes>
          <GuestCounts>
            <GuestCount AgeQualifyingCode="10" Count="1"/>
          </GuestCounts>
          <TimeSpan End="2008-01-01" Start="2008-01-08"/>
        </RoomStay>
      </RoomStays>
      <ResGuests>
        <ResGuest>
          <Profiles>
            <ProfileInfo>
              <Profile ProfileType="1">
                <Customer>
                  <PersonName>
                    <GivenName>Jan</GivenName>
                    <Surname>Novák</Surname>
                  </PersonName>
                </Customer>
              </Profile>
            </ProfileInfo>
          </Profiles>
        </ResGuest>
      </ResGuests>
    </HotelReservation>
  </HotelReservations>
</OTA_HotelResRQ>
```

Web Services

Services

- Service Oriented Architecture (SOA)
  - Service = a network component with a particular functionality which communicates using messages
  - Structure of the messages is given by the interface of the service
  - Services can exploit other services
- Advantages:
  - Interoperability – we can connect services regardless their programming languages, OS, …
  - Re-usability – we can re-use an existing service in multiple other services
  - We have an overview on the way complex services work (exploit simpler services)
  - Agility – ability to implement new requirements and changes fast and easily

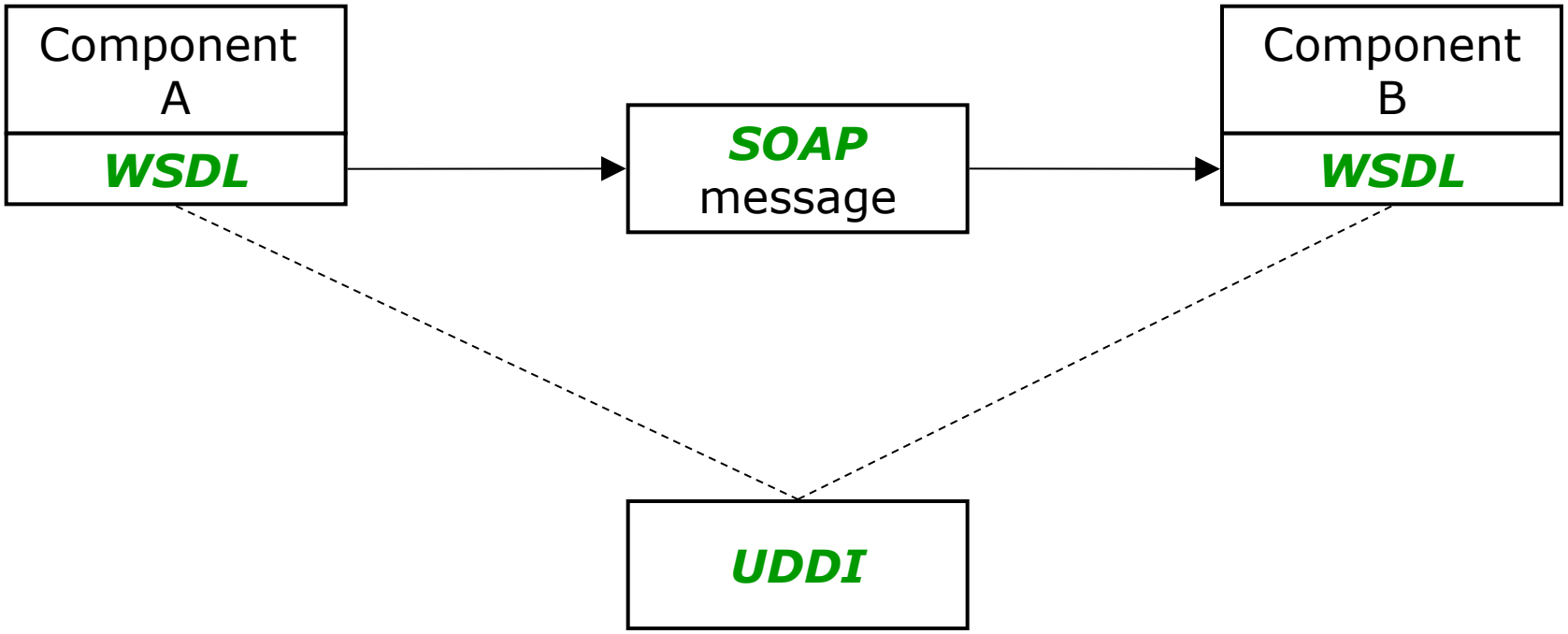
Services – Idea



Web Services

- In SOA we need:
  - To find a service that corresponds to our requirements
  - To know the structure of messages the service requires
  - To exchange the messages with the services
- We need a standard: Web Services
- Technologies of Web Services:
  - SOAP – communication among services
  - WSDL – description of service interfaces
  - UDDI – searching required services

Web Services



Simple Object Access Protocol (SOAP)

- http://www.w3.org/TR/soap12-part0/
- One of existing protocols for exchanging messages
- Advantage: platform independent
- Disadvantage: too long
  - Based on XML
- SOAP message:
  - Envelope – root element
  - Head
    - Optional, any content
    - Usually identification, authentication, …
  - Body
    - Required, content is pre-defined

SOAP – Example

Order Request

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <order-request xmlns="http://www.example.cz/schema/orders"
                   cust-number="Z001">
      <item number="V123">
        <amount>5</amount>
        <price>987</price>
      </item>
      <item number="V456">
        <amount>3</amount>
        <price>654</price>
      </item>
    </order-request>
  </env:Body>
</env:Envelope>
```

SOAP – Example

Order Response

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <order-response xmlns="http://www.example.cz/schema/orders"
                    order-number="O001"
                    expected-date="2008-01-01">
      <contact>
        <email>contact@example.cz</email>
        <phone>+420222222222</phone>
      </contact>
    </order-response>
  </env:Body>
</env:Envelope>
```

Web Services Description Language (WSDL)

- http://www.w3.org/TR/wsdl
- Description of Web Service interface
  - Which operations it offers
  - What is the input and output of each operation
- Example:
  - A Web Service offers an operation make-an-order
  - It accepts customer ID + details
    - order-request
  - It tries to add the requirement to the system
  - It successful, it returns the number of order
    - order-response

WSDL – Parameters of Operations

```
<?xml version="1.0" encoding="UTF-8"?>
<description xmlns="http://www.w3.org/ns/wsdl"
             targetNamespace="http://www.example.cz/ws/orders"
             xmlns:objsrv="http://www.example.cz/ws/orders"
             xmlns:objsch="http://www.example.cz/schema/orders"
             xmlns:wsoap="http://www.w3.org/ns/wsdl/soap"
             xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
 <types>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
               targetNamespace="http://www.example.cz/schema/orders"
               xmlns="http://www.example.cz/schema/orders">
      <xs:element name="order-request" type="OrderRequirement"/>
      <xs:element name="order-response" type="OrderResponse"/>
      ...
    </xs:schema>
  </types>
 ...
</description>
```

WSDL – Operation

```
<?xml version="1.0" encoding="UTF-8"?>
<description ... >
  <types>
    ...
  </types>

  <interface name="interface-orders">
    <operation name="make-an-order"
               pattern="http://www.w3.org/ns/wsdl/in-out">
      <input element="objsch:order-request" />
      <output element="objsch:order-response" />
    </operation>
  </interface>
  ...
</description>
```

WSDL – Way of Communication

```
<?xml version="1.0" encoding="UTF-8"?>
<description ... >
  <types> ... </types>
  <interface ...> ... </interface>

  <binding name="SOAP-making-orders"
           interface="objsrv:interface-orders"
           type="http://www.w3.org/ns/wsdl/soap"
           wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP/">
    <operation ref="tns:make-an-order"
       wsoap:mep="http://www.w3.org/2003/05/soap/mep/request-response"/>
  </binding>
  ...
</description>
```

WSDL – Service Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<description ... >
  <types> ... </types>
  <interface ... > ... </interface>
  <binding ... > ... </binding>

  <service name="service-making-orders"
           interface="objsrv:interface-orders">
    <endpoint name="endpoint-making-orders"
              binding="objsrv:SOAP-making-orders"
              address="http://www.example.cz/services/making-orders"/>
  </service>
</description>
```

Web Service Example

- ARES – Administrativní registr ekonomických subjektů
  - http://wwwinfo.mfcr.cz/ares/xml_doc/schemas/index.html

Universal Description, Discovery and Integration (UDDI)

- http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm
- A register, where service providers can register and clients can search
- Three parts:
  - White pages – basic information on providers
    - e.g. name, contact, …
  - Yellow pages – more detailed information on providers
    - e.g. its category in an industry scheme
  - Green pages – descriptions of available Web Services
    - Including WSDL
- Various types of access, including a Web Service