Methods and Tools for Management Information Systems

Lecture 1

12. Oktober 2010
Times and Dates:

Lecture: Every Tuesday, 11h15 - 12h45, room G22A-208

Discussion: Every Tuesday, 15h15 - 16h45, starting October 19, room G29-427

Office Hours:

upon appointment
Questions/Comments:

graubitz@open-source-consultants.de

Further Information (slides, exercises, etc.):

http://bauhaus.cs.uni-magdeburg.de
Literature:


R. Widhalm, T. Mück: Topic Maps. Springer, 2002 (German)

A.-W. Scheer: ARIS - Vom Geschäftsprozeß zum Anwendungssystem, Springer, 1998 (German)

Lots of online references, see webpage for details.
The course introduces technical foundations of Management Information Systems (MIS) with a strong emphasis on information representation using XML and related technologies:

- **Information representation and organisation:**
  - XML technologies: document type definition (DTD), namespaces, XML schema
  - Semantic technologies: resource description framework (RDF), RDF schema, ontologies (OWL)
  - Topic maps
  - Classification, systematics
  - Thesauri, catalogues
  - Meta data

- **Architecture of integrated information systems (ARIS):**
  - Business process (re-)engineering
  - Business process model management and improvement
  - Quality management
Extensible Markup Language (XML)

- Standard for structured, human *and* machine readable data streams
- Defines the syntax to create arbitrary yet similar markup languages (a. k. a. XML applications)
- *Idea:* strict separation between data and their representation
XML in 10 points:
“[...] And if you are giving a presentation on XML, why not start with these 10 points?” (quoted from W3C)

1. XML is for structuring data . . .

- Set of rules for designing text formats that let you structure your data
- XML makes it easy for a computer to generate data, read data, and ensure that the data structure is unambiguous
- XML is extensible, platform-independent, and it supports internationalization and localization
XML looks a bit like HTML . . .

- XML makes use of tags and attributes
- XML uses tags only to delimit pieces of data
- XML does not provide the *meaning* of the tagged data (e.g., do not assume that XML data tagged with `<p>` represents a paragraph)
- Interpretation of the data is left to the application
XML is text, but isn’t meant to be read...

- Text data allows user to inspect the files for debugging purposes
- Structure allows application to process the data automatically
- Specification explicitly forbids tolerating errors (e.g., as opposed to HTML)
XML is verbose by design . . .

- XML files are nearly always larger than comparable binary formats

- Advantages of a text format are evident (see point 3)

- Disadvantages can usually be compensated at a different level (e.g., at protocol level, application level, etc.)
5 XML is a family of technologies . . .

- The “XML family” is a growing set of modules . . .
  - XML 1.0 defines tags and attributes
  - XLink describes a standard way to add hyperlinks to an XML file
  - XPointer is a syntax in development for pointing to parts of an XML document
  - XML Schema helps developers to precisely define the structures of their own XML-based formats
  - CSS/XSL(T) for representation/transformation of XML data
  - DOM for manipulation of XML data
XML is new, but not that new . . .

- Development of XML started in 1996
- W3C Recommendation since February 1998
- Technology is not very new—before XML there was SGML
- Designers of XML . . .
  - Took the best part of SGML
  - Were guided by the experience with HTML
  - Produced something that is no less powerful than SGML, but vastly more regular and simple to use
XML leads HTML to XHTML . . .

- XHTML, the successor of HTML, is an XML application
- XHTML has many of the same elements as HTML
- Syntax has been changed slightly to conform to the rules of XML
- It inherits the syntax from XML and restricts it in certain ways
- XHTML also adds meaning to the syntax
XML is modular...

- XML allows you to define a new document format by combining and reusing other formats
- XML provides a namespace mechanism
- XML Schema is designed to support modularity at the level of defining XML document structures (e.g., it is possible to combine two schemas to produce a third one that covers a merged document structure)
XML is the basis for RDF and the Semantic Web.

- **Resource Description Framework (RDF)** is an XML text format that supports resource description and metadata applications.

- RDF provides tools to integrate more than documents, images, menu systems, and forms applications (like HTML), making the Web a little bit more into a **Semantic Web**.

- **Ontologies** as formal descriptions of terms in a certain area (e.g., shopping or manufacturing) that allow machines to communicate effectively—*like an agreement on the meanings of the words people employ in their communication*. 

---

Otto-von-Guericke-Universität Magdeburg
Arbeitsgruppe Wirtschaftsinformatik - Managementinformationssysteme - Dipl.-Kfm. Henner Graubitz
Slide 15
XML is license-free, platform-independent and well-supported . . .

- Access to a large and growing community of tools and engineers experienced in the technology
- Opting for XML is a bit like choosing SQL for databases:
  . . . you still have to build your own database and your own programs and procedures that manipulate it
  . . . but there are many tools available and many people who can help you
- Software can be built without paying license fees
- Customers are not tied to a single vendor

“[. . .] XML isn’t always the best solution, but it is always worth considering.”
XML applications consist of 3 parts:

- **Structure**
  ... defines elements, attributes, etc., which may be used in corresponding XML documents, either in form of a *document type definition* (DTD) or as an *XML schema*

- **XML documents**
  ... contain the data

- **Processing directives**
  ... define how the data is presented, e.g. using *Cascading Style Sheets* (CSS) or *Extensible Stylesheet Language* (XSL)
DTD (partial):

```xml
<!ELEMENT people ( person* )>
<!-- people is a (possibly empty) list of persons -->
<!ELEMENT person ( name, firstname, birthdate, title? )>
<!-- a person has name, firstname, birthdate and an optional title -->
<!ELEMENT name ( #PCDATA )>
<!-- name contains only text, nothing else -->
<!ELEMENT firstname ( #PCDATA )>
<!-- firstname contains only text, nothing else -->
<!ELEMENT birthdate ( #PCDATA )>
<!-- birthdate is represented as simple text -->
<!ELEMENT title ( #PCDATA )>
<!-- title contains
```
XML document (partial):

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE people SYSTEM "people.dtd">
<people>
  <person>
    <name>Miller</name>
    <firstname>Peter</firstname>
    <birthdate>12-12-1984</birthdate>
  </person>
  <person>
    <name>Smith</name>
    <firstname>John</firstname>
    <birthdate>10-10-1985</birthdate>
    <title>Dr.</title>
  </person>
</people>
```
Formatting rules (partial):

- using CSS:

```css
people { display:block; font-family:Helvetica;
  font-size:18px; font-weight:bold;
  color:green }
```
using XSL:

```xml
<xsl:template match="/">
  <html>
    <head></head>
    <body style="font-family:Helvetica; font-size:18px; font-weight:bold; color:green">
      <xsl:value-of select="." />
    </body>
  </html>
</xsl:template>
```
Formatting rules have to be referenced in the XML document, e.g.:

```xml
<?xml-stylesheet type="text/css" href="people.css"?>
```

or

```xml
<?xml-stylesheet type="text/xsl" href="people.xsl"?>
```