Georg-August-Universität
Göttingen

Applicability of XML-based Document Formats

Seminar: XML-Based Markup Languages
(WS09/10)

Simon Trang

ISO certified XML-based office formats

JTC 1/SC 34: Document description and processing languages

May 2006

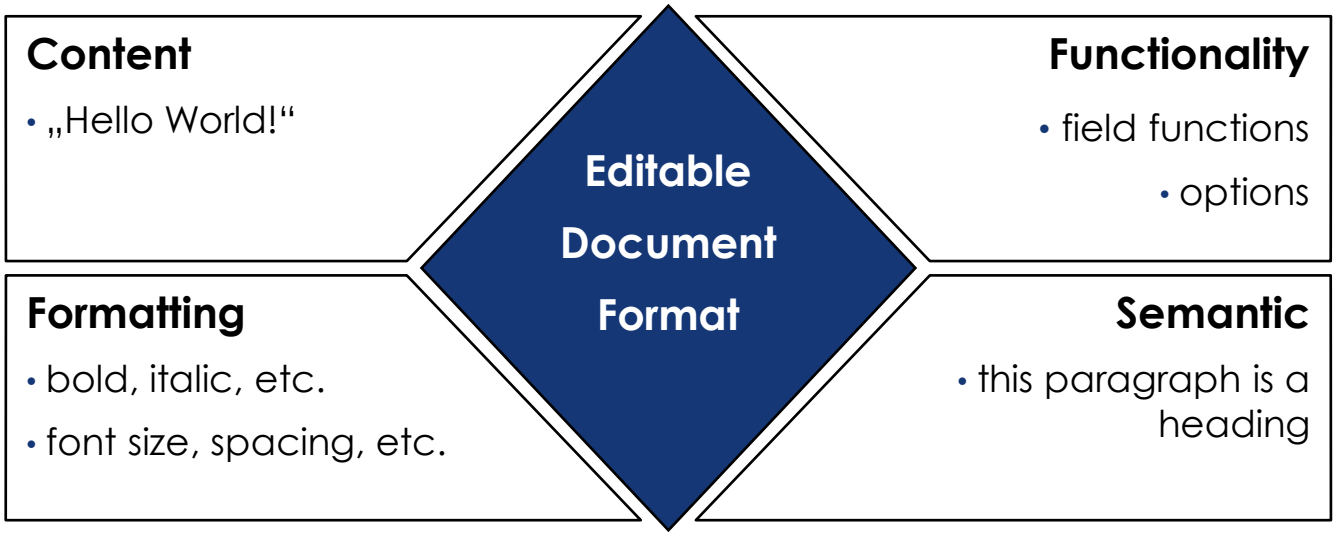
ISO/IEC 26300:2006 Open Document Format for Office Applications -- OpenDocument -- v1.0 (ODF)

November 2008

ISO/IEC 29500 Document description and processing languages -- Office Open XML File Formats -- Part 1-4 (OOXML)

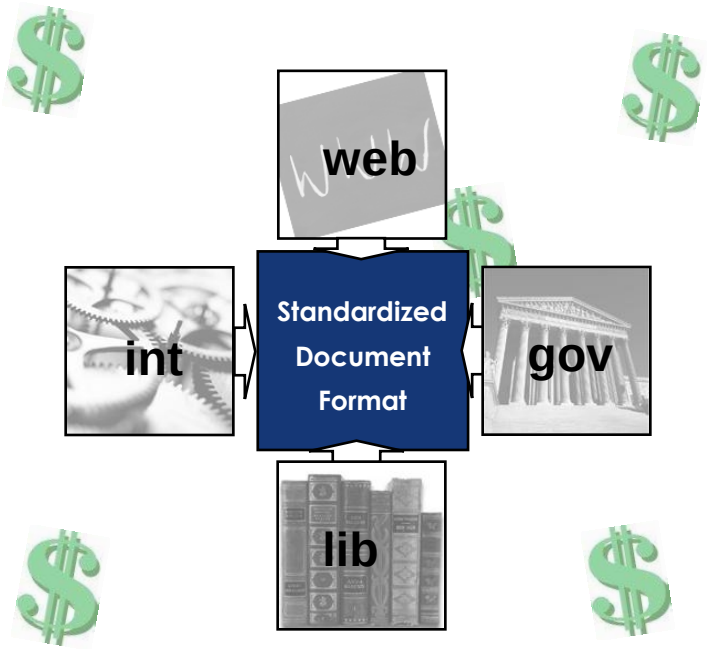
Table of Content

Challanges of editable Document Formats



Need for standardized Document Formats in XML

Drivers of standardization

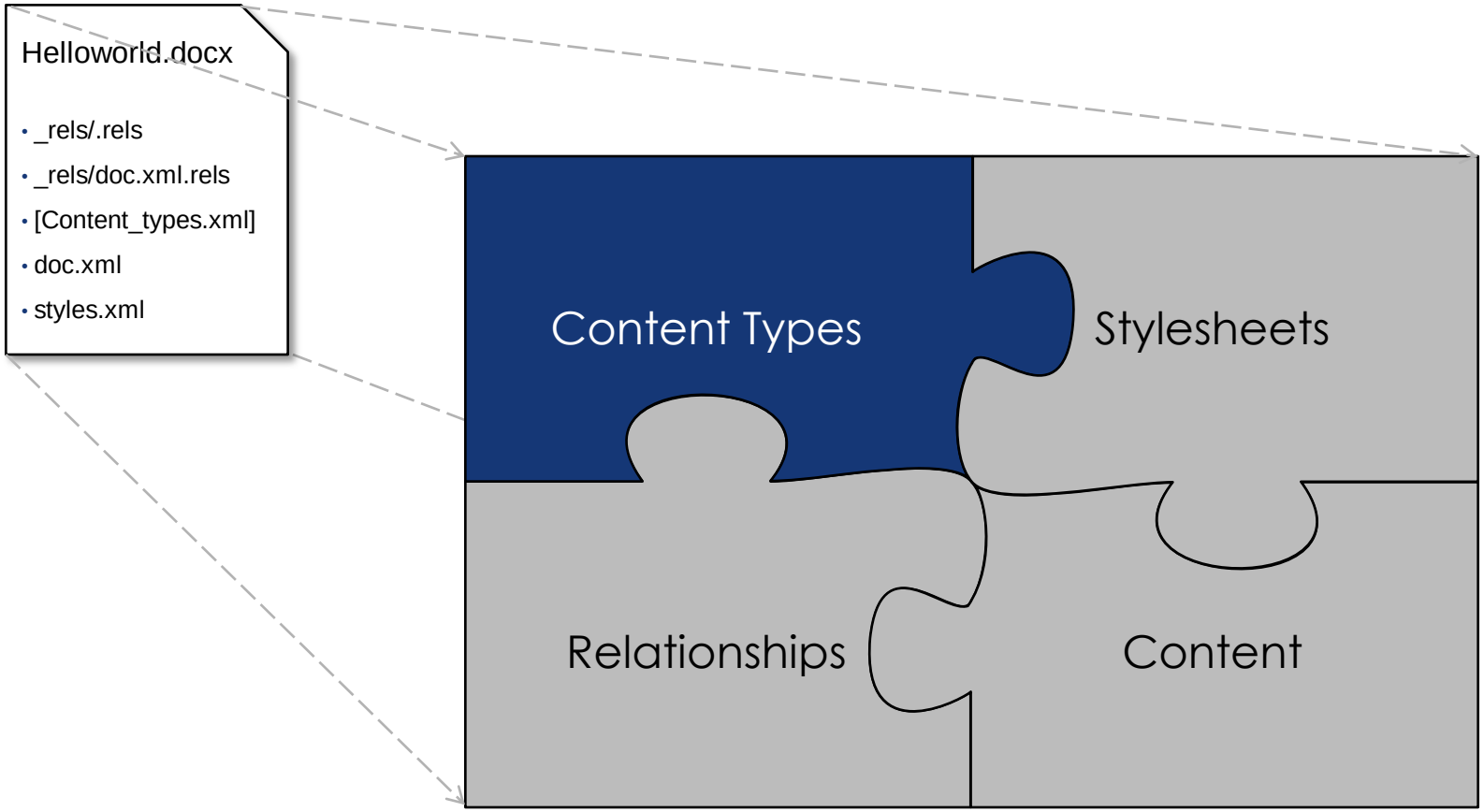


Why XML?

- Already a de-facto standard
- Easy to understand
- Tool support

Table of Content

Office Open XML: General Structure of a Container



Office Open XML: Content Types

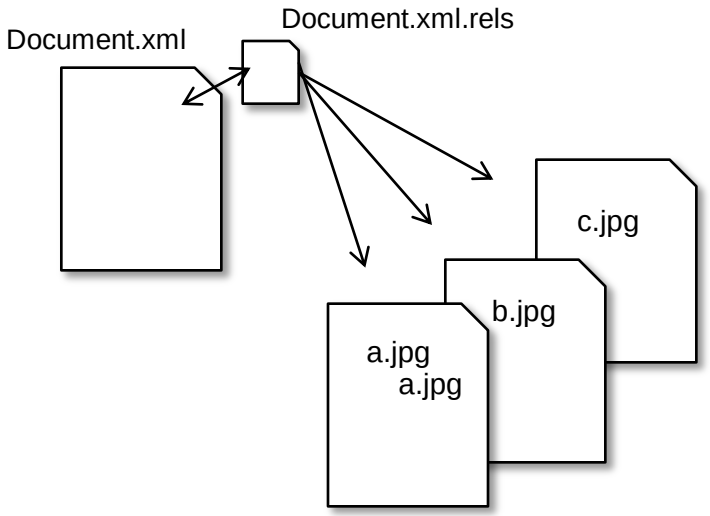
- Defines the MIME media types for all files within the container
- Helps the consumer to identify the content of a file
- Two ways to define the content type of a file:
  - a default file extension (e.g. all .png files are pngs),
  - a direct addressing (e.g. even though this files end with png, it is a text file).

A typical [Content_Types.xml]:

```
<Types>
        <Default Extension="rels" ContentType="application/vnd.openxmlformats-package.relationships+xml"/>
        <Default Extension="xml" ContentType="application/xml"/>
        <Default Extension="png" ContentType="image/png"/>
        <Override PartName="/word/document.xml" ContentType="application/vnd.openxmlformats-
officedocument.wordprocessingml.document.main+xml"/>
        <Override PartName="/word/styles.xml" ContentType="application/vnd.openxmlformats-
officedocument.wordprocessingml.styles+xml"/>
</Types>
```

Office Open XML: Relationships

- Every file can have a relationship file (filename + .rels extension)
- Every relationship has an id and carries the the path of the linked file within the package
- Relationships within a file are done by the unique identifier
- Even the entry point of the package is specified by an relationship file



A typical relationship file:

```
<Relationships>
        <Relationship Id="rId1"
        Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument"
        Target="word/document.xml"/>
</Relationships>
```

Office Open XML: Content (WordprocessingML)

- Content files have information about content and formatting properties
- In general a file with style sheets is linked

Schematic structure of a Hello World example:

```
<w:document> …
        <w:body>
                <w:p>
                        <w:pPr>
                                <w:pStyle w:val="Title"/>
                        </w:pPr>
                        <w:r>
                                <w:t>Hello World!</w:t>
                        </w:r>
                </w:p>
        </w:body>
</w:document>
```

Office Open XML: Stylesheet

- Stylesheets are linked to a content file
- They can be assigned to runs, paragraphs, tables, etc.

Schematic structure of a stylesheet file example:

```
<w:styles>
        <w:docDefaults> <w:pPrDefault> … </w:pPrDefault>
        </w:docDefaults>
        <w:style w:type="paragraph" w:styleId="Title">
                <w:name w:val="Title"/>
                <w:next w:val="StandardforContinuousText"/>
                <w:pPr>
                        <w:spacing w:after="120"/>
                        <w:jc w:val="center"/>
                </w:pPr>
                <w:rPr>
                        <w:rFonts w:ascii="Cambria"/>
                </w:rPr>
        </w:style>
</w:styles>
```

Open Document Format: General Container Structure

Style classes

- ODF has a fixed packet structure
- Similar to the Content Types of OOXML the manifest.xml describes the type of each file
- The mimetype shows assessing applications the document type
- The meta.xml has meta information about the document like author, date, etc.

Helloworld.odf

- _META-INF/manifest.xml
- content.xml
- meta.xml
- mimetype
- styles.xml
- …

Open Document Format: Styles

Style classes

- Common styles

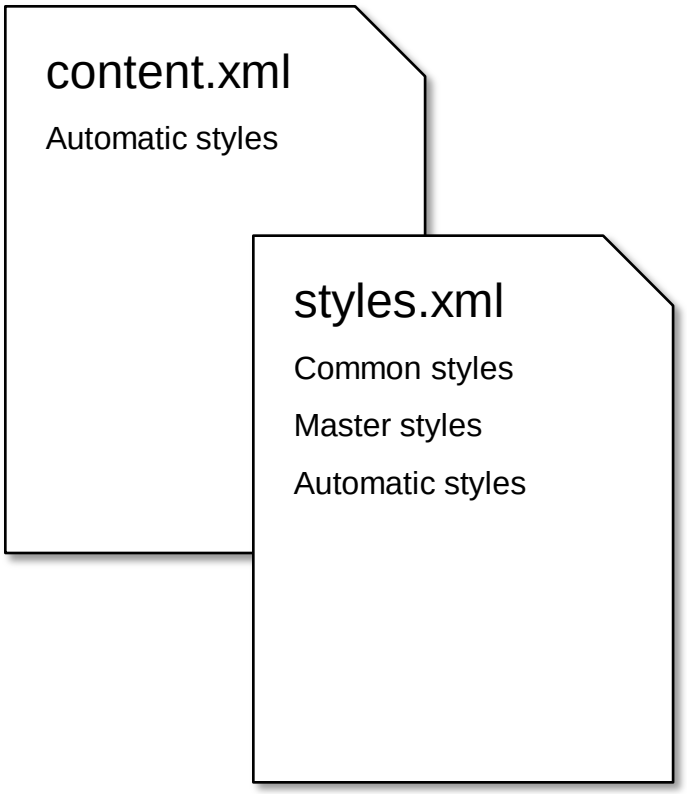
(User-)Predefined style information like paragraph or page formatting

- Master styles

Extend Common styles with additional content like header or footer

- Automatic styles

Automatic defined styles, especially for formatting without user defined styles



Open Document Format: document.xml

```
<office:document-content office:version="1.0">
        <office:automatic-styles>
                <style:style style:name="p1" style:family="paragraph" style:parent-style-name="Standard">
                        <style:paragraph-properties fo:text-align="center"/>
                </style:style>
        </office:automatic-styles>
        <office:body>
                <office:text>
                        <text:p text:style-name=“p1">Hello World!</text:p>
                </office:text>
        </office:body>
</office:document-content>
```

Evaluation of OOXML and ODF

OOXML

- Long syntax
- Mixture of content and formatting properties
- File access with relationships
- Little use of existing standards
- A lot of tool support, e.g. APIs for programming languages

ODF

- Short syntax
- Sharp distinction between content and layout
- Fixed packet structure
- Reuse of existing standards like xsl:fo, XLink and SVG

Table of Content

Idea of a wordprocessing-based Content Management

Content Management System

- Supports easy data storage and retrieval
- Often, manual data input is immature

Wordprocessing Application

- User-friendly and sophisticated
- Macros allow additional functionality

Extensible Stylesheet Language

- Stream-based language to transform XML

Wordprocessing Documents as data input

- Prepared wordprocessing files have input fields
- Fields can be tagged with styles
- Macro can check validation
- XSL script transforms document to a format as requested (e.g. DB)
- Simple formatting properties like bold or italic can be used

Application on university context: updating Modules

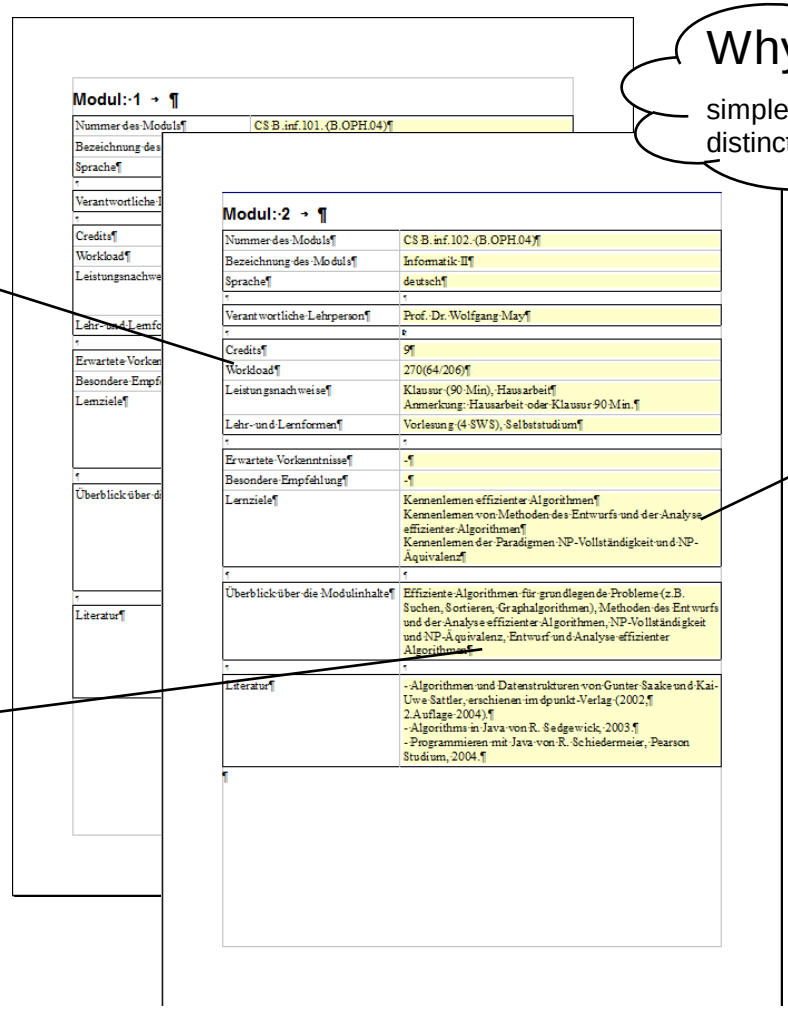
Forward forms via mail

Send filled forms back

Content Management System

- Initialize module updating
- Generate and send forms
- Wait for answers
- Check and process forms
- etc.

Implementation in ODF (Open Office)



Why ODF?

simple syntax and strict distinction of style and content

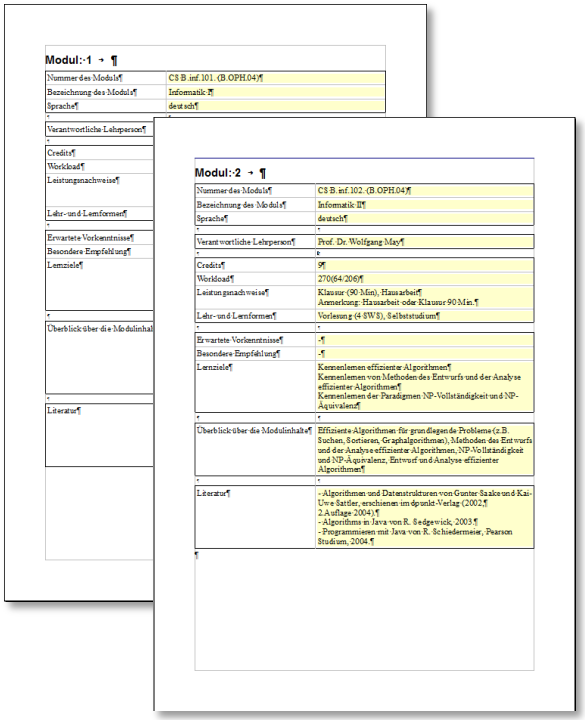
Field description (is not relevant for processing)

Stylesheets indicate text fields

Yellow background color indicates that a style is assigned

Comments on XSLT Transformation

content.xml

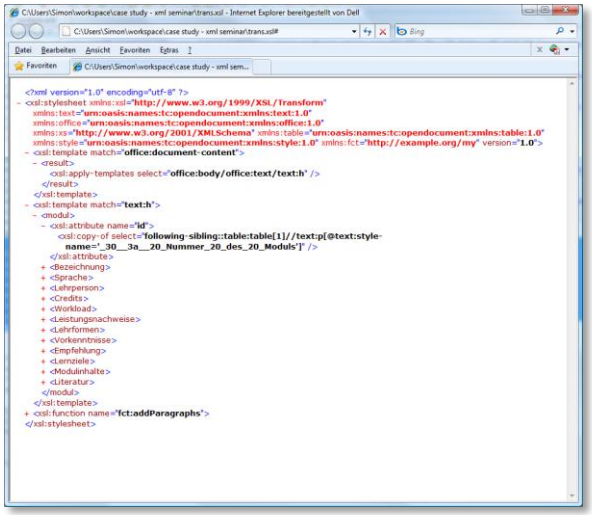


Modul:-1 → ¶

Nummer des Moduls¶	CS B.inf.101. (B.OPH.04)¶
Bezeichnung des Moduls¶	Informatik-I¶
Sprache¶	deutsch¶
Verantwortliche Lehrperson¶	
Credits¶	
Workload¶	
Leistungsnachweise¶	
Lehr- und Lernformen¶	
Erwartete Vorkenntnisse¶	
Besondere Empfehlung¶	
Lernziele¶	
Überblick über die Modulinhal	
Literatur¶	

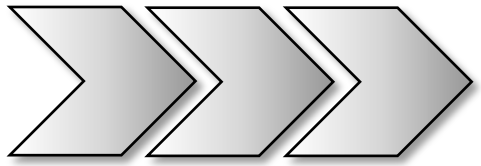
Modul:-2 → ¶

Nummer des Moduls¶	CS B.inf.102. (B.OPH.04¶
Bezeichnung des Moduls¶	Informatik-II¶
Sprache¶	deutsch¶
Verantwortliche Lehrperson¶	Prof. Dr. Wolfgang May¶
Credits¶	9¶
Workload¶	270(64/206)¶
Leistungsnachweise¶	Klausur (90 Min), Hausarbeit¶ Anmerkung: Hausarbeit oder Klausur 90 Min.¶
Lehr- und Lernformen¶	Vorlesung (4 SWS), Selbststudium¶
Erwartete Vorkenntnisse¶	-¶
Besondere Empfehlung¶	-¶
Lernziele¶	Kennenlernen effizienter Algorithmen¶ Kennenlernen von Methoden des Entwurfs und der Analyse effizienter Algorithmen¶ Kennenlernen der Paradigmen NP-Vollständigkeit und NP-Äquivalenz¶
Überblick über die Modulinhalte¶	Effiziente Algorithmen für grundlegende Probleme (z.B. Suchen, Sortieren, Graphalgorithmen), Methoden des Entwurfs und der Analyse effizienter Algorithmen, NP-Vollständigkeit und NP-Äquivalenz, Entwurf und Analyse effizienter Algorithmen¶
Literatur¶	- Algorithmen und Datenstrukturen von Gunter Saake und Kai-Uwe Sattler, erschienen im dpunkt-Verlag (2002.¶ 2.Auflage 2004)¶ - Algorithms in Java von R. Sedgewick, 2003.¶ - Programmieren mit Java von R. Schiedermeier, Pearson Studium, 2004.¶



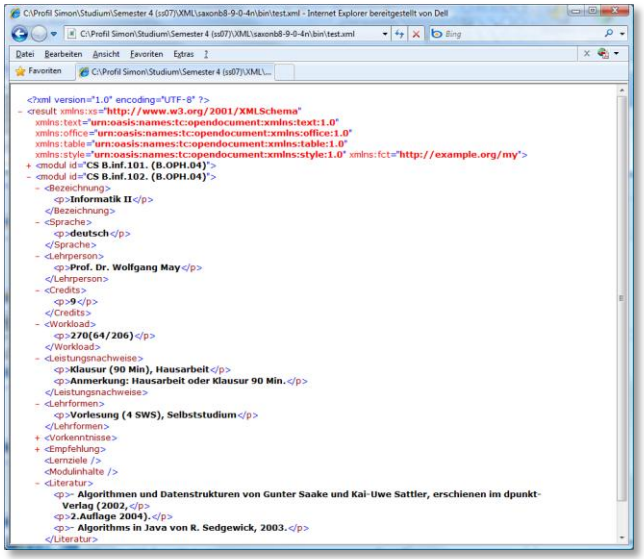
stylesheet.xsl

```
<?xml version="1.0" encoding="utf-8" ?>
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:text="urn:oasis:names:tc:opendocument:xmlns:text:1.0"
    xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:table="urn:oasis:names:tc:opendocument:xmlns:table:1.0"
    xmlns:style="urn:oasis:names:tc:opendocument:xmlns:style:1.0" xmlns:fct="http://example.org/my" version="1.0">
  - <xsl:template match="office:document-content">
    - <result>
        <xsl:apply-templates select="office:body/office:text/text:h" />
      </result>
    </xsl:template>
  - <xsl:template match="text:h">
    - <modul>
      - <xsl:attribute name="id">
          <xsl:copy-of select="following-sibling::table:table[1]//text:p[@text:style-
            name='__30__3a__20_Nummer_20_des_20_Moduls']" />
        </xsl:attribute>
      + <Bezeichnung>
      + <Sprache>
      + <Lehrperson>
      + <Credits>
      + <Workload>
      + <Leistungsnachweise>
      + <Lehrformen>
      + <Vorkenntnisse>
      + <Empfehlung>
      + <Lernziele>
      + <Modulinhalte>
      + <Literatur>
      </modul>
    </xsl:template>
  + <xsl:function name="fct:addParagraphs">
  </xsl:stylesheet>
```



XSL Processor

result.xml



```
<?xml version="1.0" encoding="UTF-8" ?>
- <result xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:text="urn:oasis:names:tc:opendocument:xmlns:text:1.0"
    xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
    xmlns:table="urn:oasis:names:tc:opendocument:xmlns:table:1.0"
    xmlns:style="urn:oasis:names:tc:opendocument:xmlns:style:1.0" xmlns:fct="http://example.org/my">
  + <modul id="CS B.inf.101. (B.OPH.04)">
  - <modul id="CS B.inf.102. (B.OPH.04)">
    - <Bezeichnung>
        <p>Informatik II</p>
      </Bezeichnung>
    - <Sprache>
        <p>deutsch</p>
      </Sprache>
    - <Lehrperson>
        <p>Prof. Dr. Wolfgang May</p>
      </Lehrperson>
    - <Credits>
        <p>9</p>
      </Credits>
    - <Workload>
        <p>270(64/206)</p>
      </Workload>
    - <Leistungsnachweise>
        <p>Klausur (90 Min), Hausarbeit</p>
        <p>Anmerkung: Hausarbeit oder Klausur 90 Min.</p>
      </Leistungsnachweise>
    - <Lehrformen>
        <p>Vorlesung (4 SWS), Selbststudium</p>
      </Lehrformen>
    + <Vorkenntnisse>
    + <Empfehlung>
      <Lernziele />
      <Modulinhalte />
    - <Literatur>
        <p>- Algorithmen und Datenstrukturen von Gunter Saake und Kai-Uwe Sattler, erschienen im dpunkt-
          Verlag (2002,</p>
        <p>2.Auflage 2004).</p>
        <p>- Algorithms in Java von R. Sedgewick, 2003.</p>
      </Literatur>
```

Evalutation

Advantages

- Wordprocessing applications are very user-friendly
- Everyone can handle them
- They are widely spread and some are for free
- Software is mature
- You can use features like spell- and grammerchecker
- Simple formatting can be used
- Macros are a powerful tool to enhance functionality (e.g. check constraints, …)

Limits and challanges

- Objects like paragraphs or cells can only be tagged once
- Even though the documents are written in XML, the heading tags are sequential
- The work on document is offline

Conclusion and Outlook

- New XML-based document formats offer easy access on document content
- Tagged content can be processed very easily to extract content
- Some difficulties with non-hierarchical heading structure
- Import of content from ODF files general possible

- Scenario should be extended
- Until now, macros are not implemented
- Generic framework for content extraction with stylesheets is imaginable
- This pattern can be applied to many other problems

Thank you for you attention!

Questions?

Backup

History of XML-based office formats