

Using OpenOffice for ETDs

Workshop, ETD 2003

Volker John

Computing Center, University of Hamburg



Today's Topics

- > ETD Requirements
- > Some Pre-requisites
- > Preparing OpenOffice
- > Using OpenOffice to create XML ETDs
- > Strengths and Weaknesses
- > Conclusion

Some Pre-Requisites for this Session

- > Limited, general XML knowledge (will be provided during this session)
- > A small familiarity with OpenOffice is beneficial (or at least working knowledge of a GUI-Wordprocessor)
- > For your own ETD design: XSLT knowledge
 - will not be provided, but necessary topics for OpenOffice integration will be covered

Apply Structures using XML

A very brief XML Introduction

Basic Ideas

> Markup Languages

- Identifikation of Information Objects by means of tags

> Machine-readable or understandable for persons?

> Abstract versus operational semantics

> DTD - Document Type Definition

- Description of classes of information and those hierarchical relationships allowed between specific information objects

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE product SYSTEM
"http://www.some.com/DTDs/product.dtd">
<product id="prod0815">
  <name>candlelight</name>
  <price>15.80</price>
  <currency name="EUR"/>
  <mf>&manufacturer;</mf>
</product>
```

```
<!DOCTYPE product [
<!ELEMENT product (name, price, currency, mf)>
<!ATTLIST product id ID #REQUIRED>
<!ELEMENT name (#PCDATA)>
<!ELEMENT price (#PCDATA)>
<!ELEMENT currency EMPTY>
<!ATTLIST currency name (EUR|DM) "DM">
<!ELEMENT mf (#PCDATA)>
<!ENTITY manufacturer „Paraffin GmbH">
]>
```

> Document Type Definition (DTD)

- Specialized syntax to define the structure of XML documents
- Document Type Definitions (DTDs), XML Schema Languages
- Describe allowed set and structure of tags
- Programmatic checks of DTD/Schema compliance is possible

> XML Documents

- Markup
 - `<elementName> ... </elementName>`
 - `<elementName/>`
 - `<elementName attribut=„Value“> ... </elementName>`
- Processing Instructions
 - `<? objectID=(xxx-yyy-zzz) ?>`
- Comments
 - `<!-- this is a comment -->`
- Entities
 - `&manufacturer;`
- but... no layout nor presentation information!

Characteristics

> Tree Structures in XML

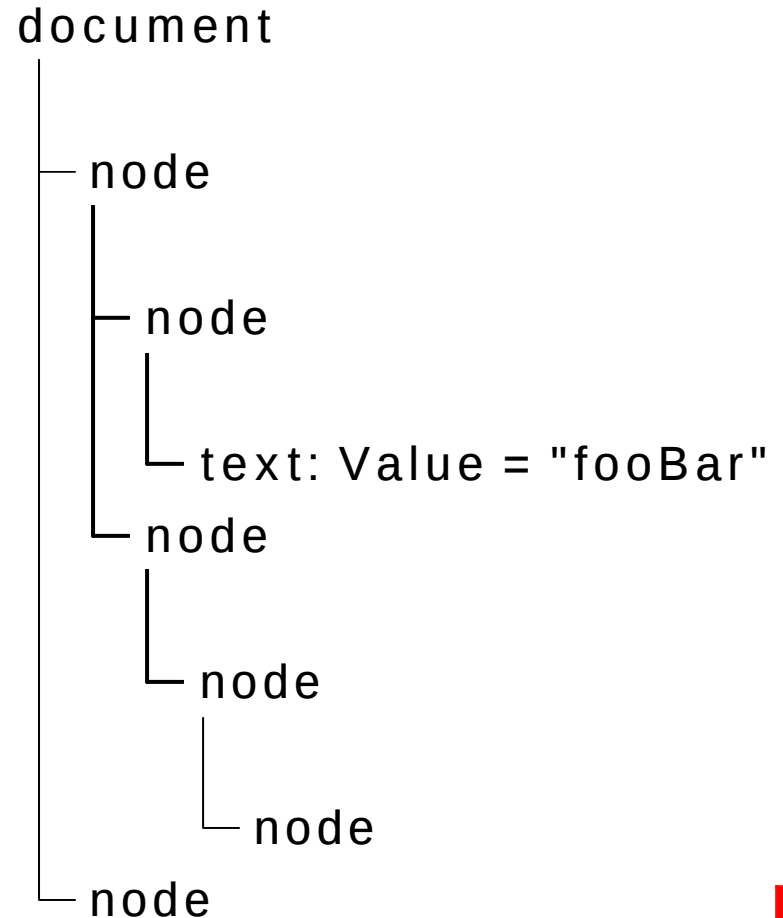
- Note: Tree structures are easily modeled in object-oriented applications

> Platform-independent

- ASCII
- UNICODE

> Definition allows for validity checks using machine „intelligence“

> Valid versus well-formed XML documents



Structured Documents

> Advantages

- Long-lived
- Easy re-use and re-purposing
- Automated processing
- Added efficiencies for large instances
- Single source, multiple targets

> Applications

- Technical Documentation
- Scientific Publishing
- Application Messaging, Data Exchange

OpenOffice Setup Woes

Preparing OpenOffice

> Java 1.4 JRE Installation

- Developers could also use JDK 1.4

> Download and install OpenOffice Beta

- Make sure to install mobile device filters

> Enable OpenOffice XML Docbook support

- See <http://xml.openoffice.org/xmerge>

> Possibly get XML Docbook

- Docbook support in OpenOffice defaults to http-based system identifier
- <http://www.oasis-open.org/docbook/xml/>

> Stylesheets for XML Docbook

- Obtained from SourceForge

Get & Install JRE 1.4

- > Sun JRE Distribution

- <http://java.sun.com/j2se/1.4.1/download.html>

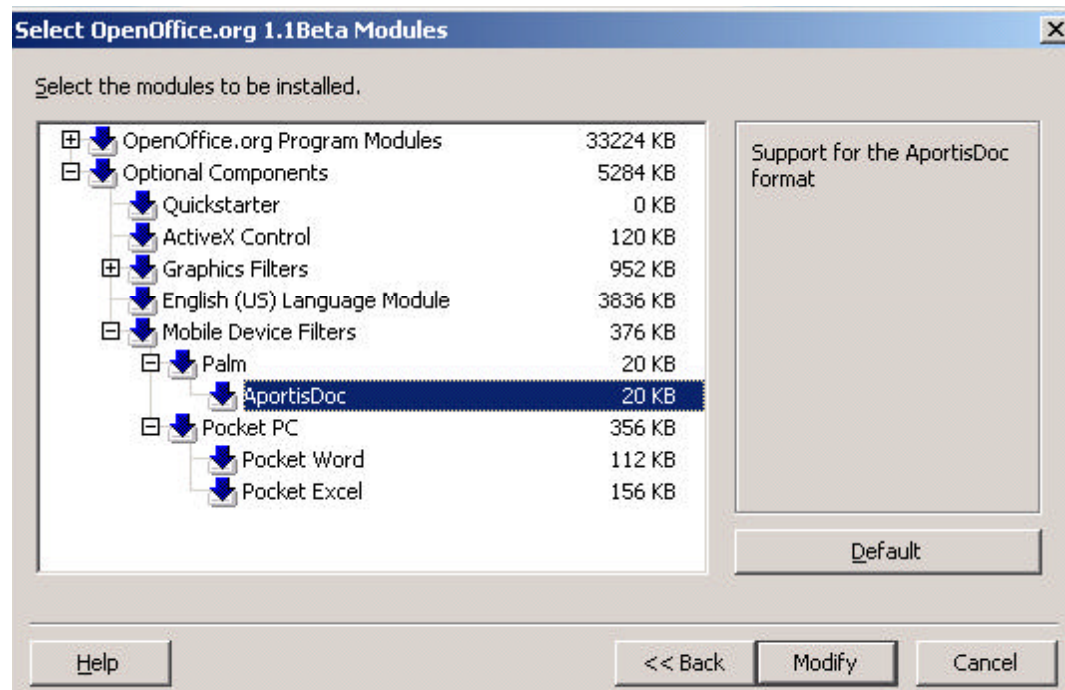
- > Contains Crimson XML Parser & Xalan XSLT Processor

- > Required for OpenOffice Installation & Execution of Java-code within OpenOffice

OpenOffice Setup

> Download & Install OpenOffice

- make sure to get version 1.1 / currently in beta
- available from www.openoffice.org
- Make sure to install mobile device filters



Required for OpenOffice XML Docbook Support

> Document Editing on Small Devices – XMerge

- Aimed at small devices, can be used for general transformation to / from OpenOffice XML file format

> You will also need a so-called “file format filter”

- Download from xml.openoffice.org/xmerge/docbook/
- Note: this requirement does no longer apply for OO1.1B2, available soon
- Side note: there is one available for OpenOffice Format to LaTeX, too

> Get a copy of the Docbook template

- Download from http://xml.openoffice.org/xmerge/downloads/docbook_template.sxw
- Install to <OpenOffice Installation>\share\template\<language>[...] to have it listed as a template
- Note: this requirement does no longer apply for OO1.1B2, available soon; the Docbook template is part of the distribution

Setting up OpenOffice to create XML ETDs

> TypeDetection.xc

- Located at <OpenOffice Installation>\share\registry\data\org\openoffice\Office
- Open using Text Editor
- Search for <node oor:name="DocBook File" oor:op="replace">
- Before corresponding closing tag </node> insert

```
<prop oor:name="Installed" oor:type="xs:boolean">  
  <value>true</value>  
</prop>
```

- Similarly for Flat XML (this is for the advanced part of this session)
- Note: this requirement does no longer apply for OO1.1B2, available soon

On the Setup Side,

We are done.

The Authoring Side

Using Docbook as intended Target



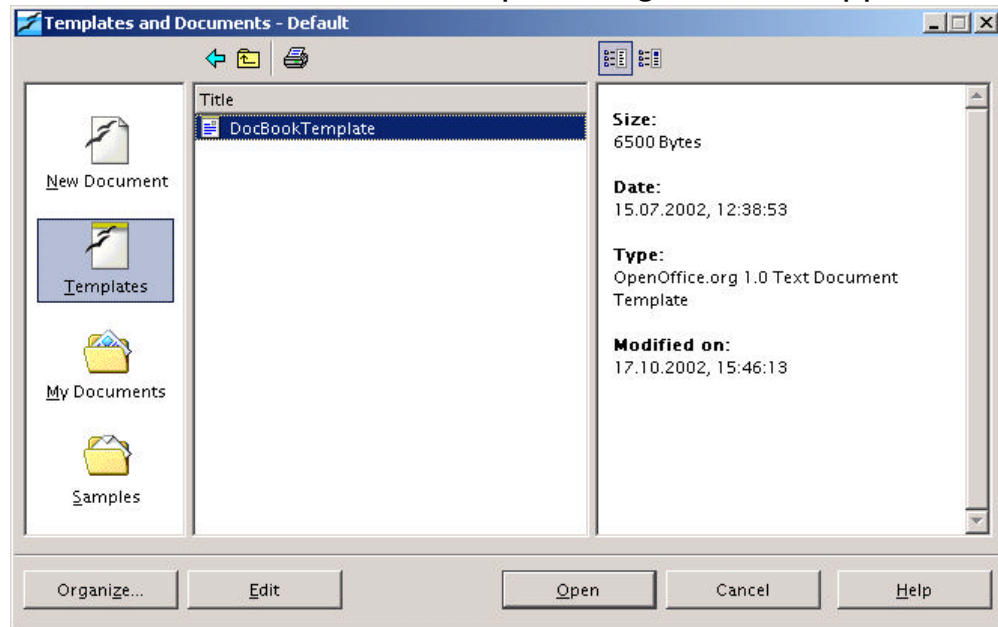
Creating your first Docbook Document

> Start OpenOffice by selecting

- Start Menu
- Programs
- OpenOffice[...] (depends on version)
- From Template

> Select DocBook Template

- When placed into <OpenOffice Installation>\share\template\english, it will appear as shown
- Open the Template



What is the trick behind OpenOffice's XML capabilities?

- > OpenOffice Sections, Tables, and Styles are mapped to Docbook sections and elements
- > OpenOffice uses an XML file format internally
 - This can easily be transformed to other XML dialects
- > OpenOffice does not validate the document structure
 - Saving lots of implementation efforts
 - Possibly creating some problems with transformations of resulting documents to other formats
- > Only a limited set of Docbook elements currently supported
 - See <http://xml.openoffice.org/xmerge/docbook/DocBookTags.html>

Sample Docbook Usage

- > Once a document based on the Docbook template is created, it already contains two sections:
 - The document info section (this is where the document title goes)
 - A first section (you can easily type ahead)
- > Create the Document Title
 - Will be mapped to article title
 - Create other document matter
- > Enter information into the 1st Section
 - Simply type the section title, then hit return to enter the section's body
 - Note that *Text Body* style is mapped to Docbook's *para* element
- > Create more sections
 - Choose Insert | Section from menu
 - Each section requires a unique name ("New Section")
 - Set section title and enter body text
 - Note: for nested sections, it proves useful to have the navigator open (F5)
 - Note: to leave a section, hit ALT-RETURN

Extended Docbook Usage

> Tables

- Select Insert | Table vom Menu Bar
- Create a table title by adding a caption (right click on table, then choose caption)

> Character styles

- From Style List Dialog, choose character styles
- Mark selection, then choose appropriate character style (e.g., Emphasis, Filename, Command, ...)

> Images

- Be sure to check the Link checkbox the import dialog, otherwise, the fileref attribute of the inlinegraphic element will remain empty

> Formulas

- Currently not supported

The Customization View

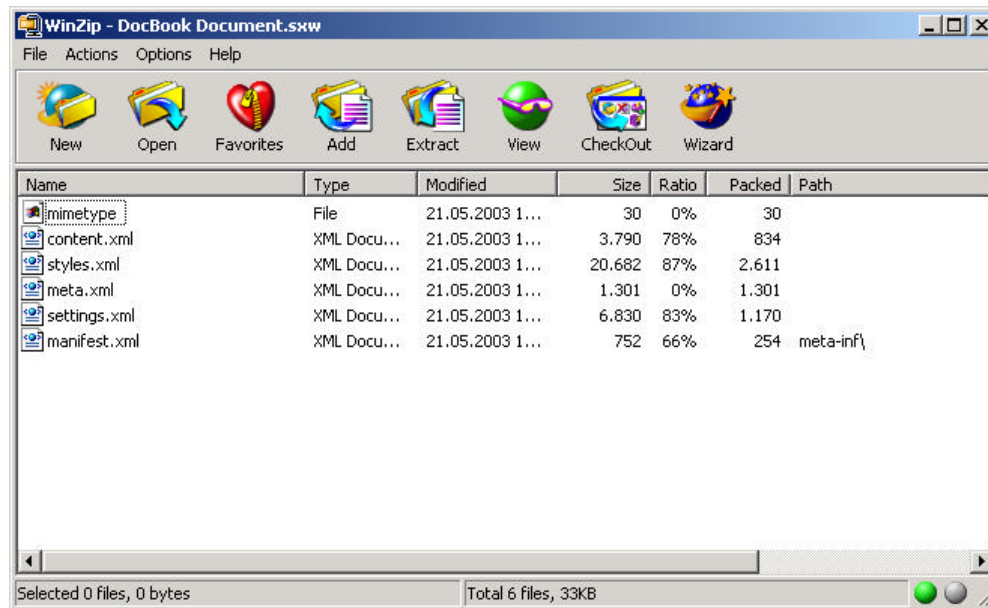
Creating other XML Documents

Using OpenOffice to create XML ETDs

- > Note: the following applies only to OpenOffice 1.1 Beta 2 (and newer)
 - Additional information: XML filters are installed as part of the full installation (or by custom choice)
- > The previous lessons mostly dealt with creating Docbook XML documents, to give some insights on handling
- > The next steps will show you how to create XML structured according to your own DTD from within OpenOffice

Some .SXW Knowledge

- > .SXW is the file extension that OpenOffice Writer documents use
- > These files are .ZIP archives
 - Try opening them with WinZIP (or similar) for a start!



- Content.xml is our point of interest

What you need is...

> ...an XSLT Stylesheet

- XSLT is 'eXtensible Stylesheet Language Transformations'
 - See <http://www.w3.org/Style/XSL/> for further details
- XSLT is a language described as an XML DTD
 - Thus, all XSLT documents are XML documents aiming to describe how an XML document can be transformed into another
 - The above is what makes XSLT so useful in conjunction with OpenOffice
- XSLT will be used to map content.xml styles to your own ETD DTD elements
- Note: OpenOffice allows roundtripping – thus, you will eventually need two stylesheets; one for export, the other for import

> ...and knowledge about the structure of OpenOffice content.xml files

- You do not need to know about programming OpenOffice using C++ and / or Java!

> ...to describe your own XML output format

User Quick Guide

> Some easy steps

- Study Content XML Structure
- Create and test stylesheet (at least the one required for export) for your ETD
- Create new filter using OpenOffice dialog
 - Located under Tools | XML Filter Settings...
- Choose File | Save As...
- Done

> Notes

- You also require your own ETD DTD to supply to OpenOffice
- Probably the most efforts will go into the creation of the stylesheet

> XML filters in OpenOffice use the XMerge framework's XSLT processing functionality

- New transformations can be created using the steps provided before
- Exactly, it is required to create a .JAR (Java Archive) file with the following contents:
 - a set of two XSLT style-sheets, one for transforming from your ETD to OpenOffice and one for transforming from OpenOffice to your ETD.
 - A file called converter.xml file in the META-INF directory that contains information describing the supported mime-types, the style-sheet names and the XMerge plugin that your ETD transformation uses

```
<converters>
<converter type="staroffice/sxw" version="1.0">
<converter-display-name>
[...]
<converter-class-impl>
org.openoffice.xmerge.converter.xml.xslt.PluginFactoryImpl
</converter-class-impl>
[...]
</converter>
</converters>
```
- All of the above information can be managed either using the OpenOffice menu items (previous slide) or manually by authoring the appropriate files

Something to Take Home

Strengths and Weaknesses

> Pros

- Freely available
- Standardized document type with long history ensures high interoperability and excellent potential for preservation
- Commercial “offspring” Star Office distributed freely among schools and other educational institutions
- Works among almost all platforms

> Cons

- Docbook support still limited
- Docbook- related styles are not clearly identified (relation is not self-explaining)
- Each other DTD requires additional coding
- OpenOffice does not restrict users from breaking the template

Food for Thoughts

> Rant

- Why should I use OpenOffice/XML at all? Going from OpenOffice to Word/HTML/PDF works fine for me!

> Suggested Reading

- Save as XDiML (DissertationMarkupLanguage), Writing and Converting digital Theses and Dissertations using OpenOffice
<http://marketing.openoffice.org/conference/presentations-pdf/thu1615/XDIML.pdf>

Questions & Answers
