The World of ODF

Only a little bit more
than you need to know

Rob Weir
IBM
robert_weir@us.ibm.com
http://www.robweir.com/blog

Tshwane
June, 2008

Outline

- What is ODF?
- History of document formats
- OASIS and the ODF TC
- Standardization of ODF
- Packaging Model
- Accessibility
- Metadata
- Extending ODF
- Toolkits/Libraries
- ODF Interoperability

What is ODF?

Specification Coverage

- Storage formats for:
  - Text documents
  - Spreadsheet documents
  - Presentation documents
- And the templates for the above
- Interesting subsets:
  - Drawings
  - Charts

Out of Scope

- Database (OpenOffice Base) format
- Clipboard formats
- Runtime API's as used by scripts
- Scripting languages
- User Interface

RELAX NG Schema

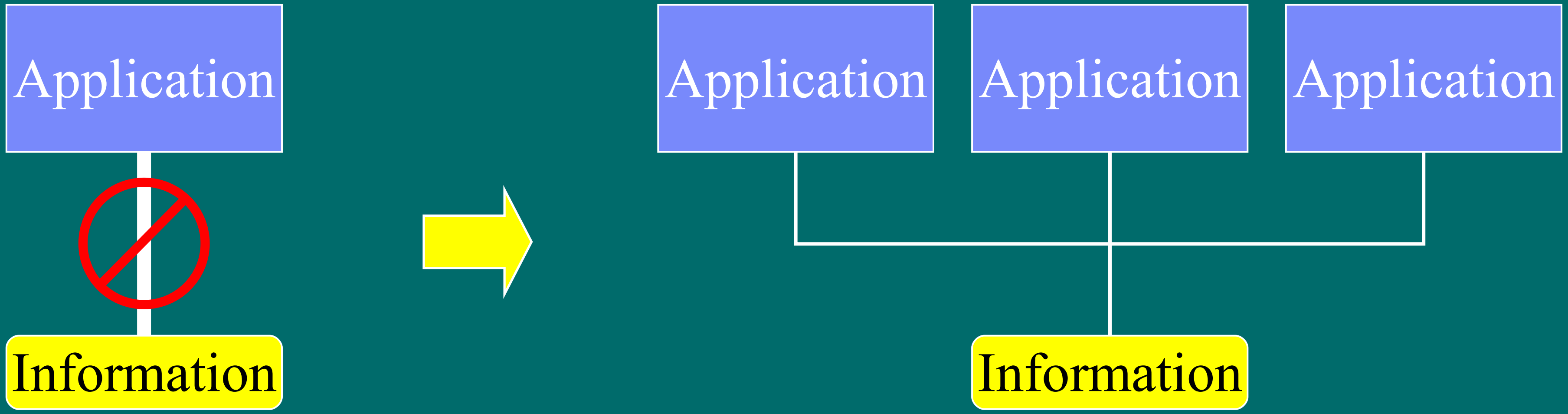
- Main schema defined in RELAX NG
- But also embedded markups defined:
  - In XML Schema, e.g., MathML 2.0, XForms 1.0
  - In DTD, e.g., XLink

- This makes validation more complicated

- Namespace-based Validation Dispatching Language (ISO NVDL) can help here

Why change? Why Now?

ODF is an example of a bigger change



Old Style

Information is closely linked to the application that created it.

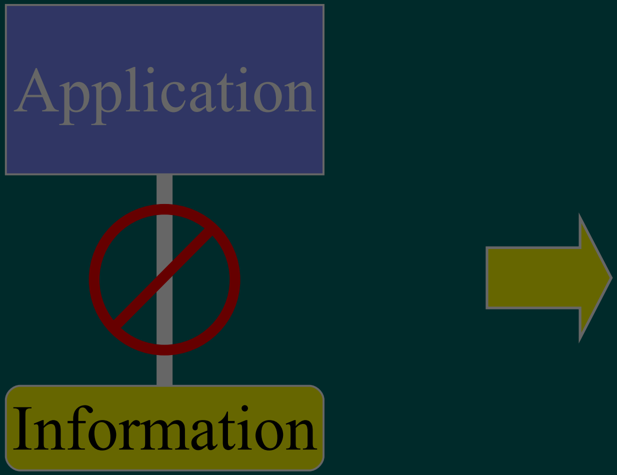
Control is with the software developer *not* the customer.

New Style

Information is represented using a real open standard not under the control of a single vendor, and multiple applications can create and access it interchangeably.

Control is with the customer *not* the software provider.

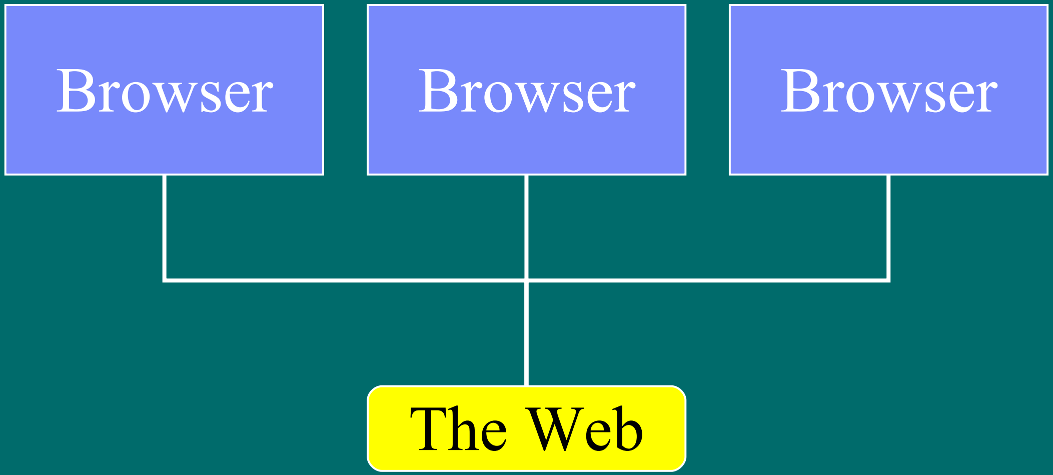
We started to see this in the 1990s



Old Style

Information is closely linked to the application that created it.

Control is with the software developer *not* the customer.

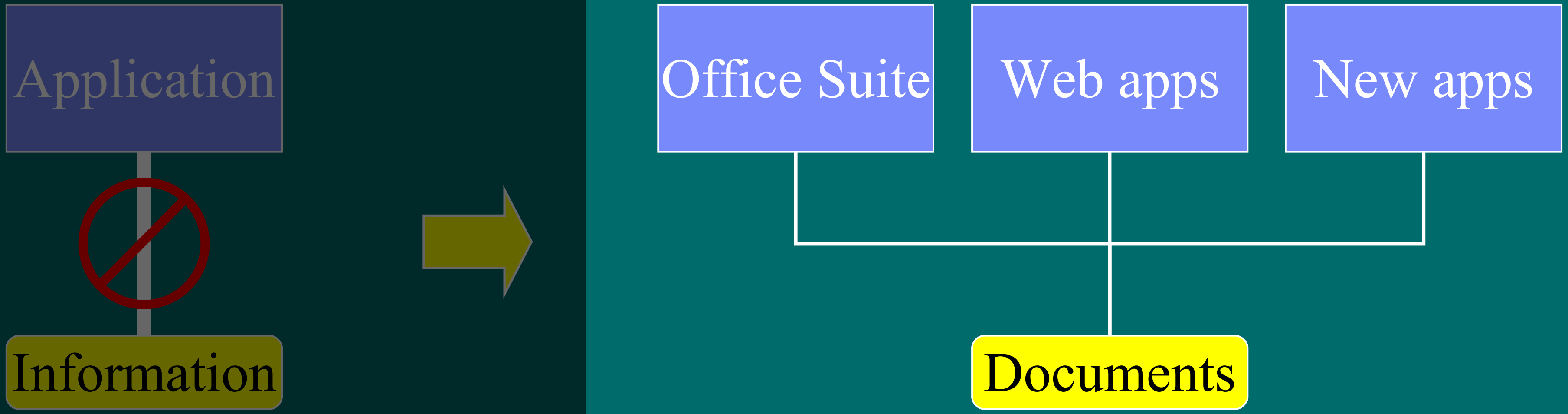


New Style

Information is represented using a real open standard not under the control of a single vendor, and multiple applications can create and access it interchangeably.

Control is with the customer *not* the software provider.

The trend will accelerate in the 2000s



Old Style

Information is closely linked to the application that created it.

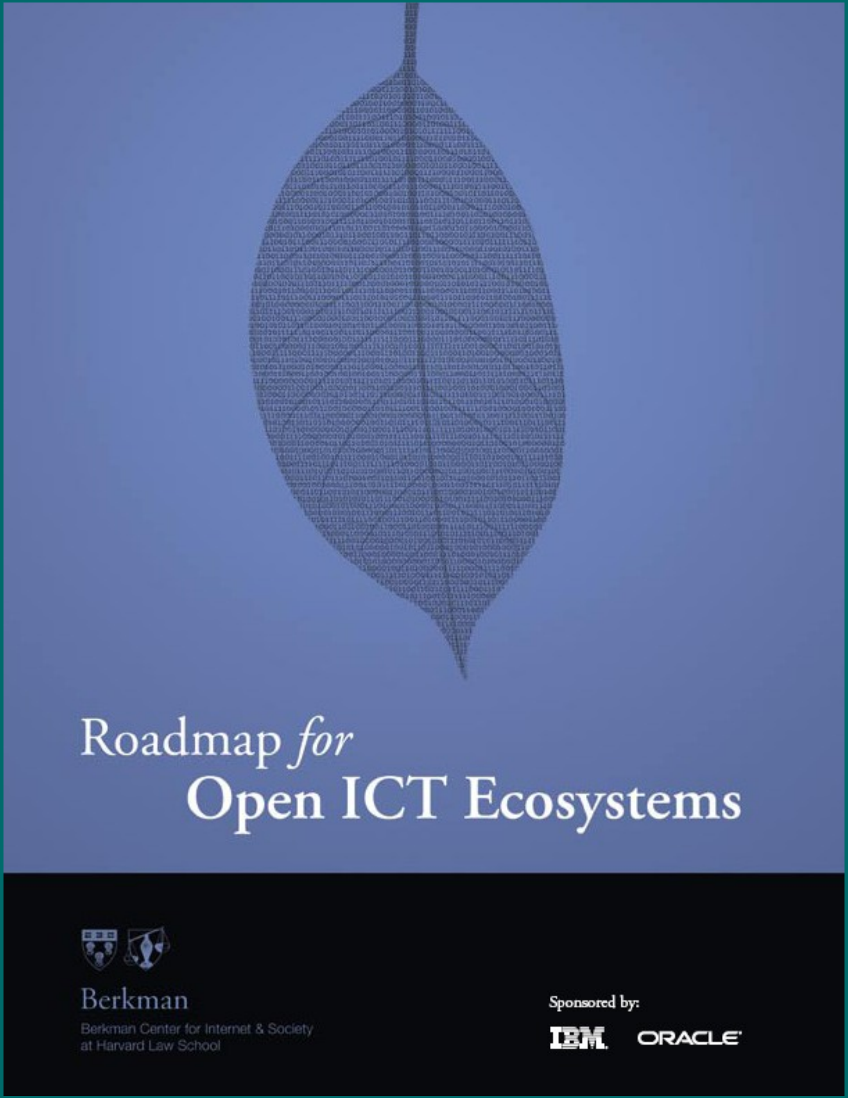
Control is with the software developer *not* the customer.

New Style

Information is represented using a real open standard not under the control of a single vendor, and multiple applications can create and access it interchangeably.

Control is with the customer *not* the software provider.

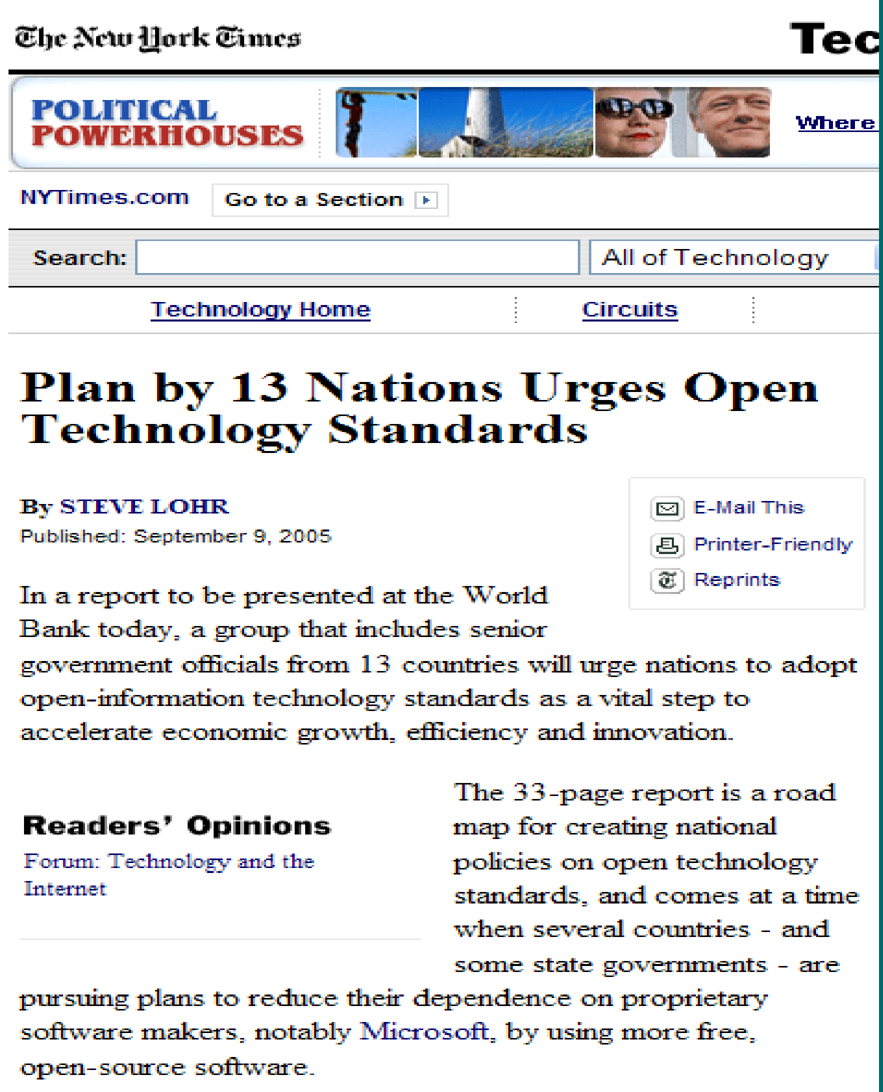
This is part of the bigger "Open" movement



Roadmap *for*
Open ICT Ecosystems

Berkman
Berkman Center for Internet & Society
at Harvard Law School

Sponsored by:
IBM ORACLE



Plan by 13 Nations Urges Open Technology Standards

By STEVE LOHR
Published: September 9, 2005

In a report to be presented at the World Bank today, a group that includes senior government officials from 13 countries will urge nations to adopt open-information technology standards as a vital step to accelerate economic growth, efficiency and innovation.

Readers' Opinions
Forum: Technology and the Internet

The 33-page report is a road map for creating national policies on open technology standards, and comes at a time when several countries - and some state governments - are pursuing plans to reduce their dependence on proprietary software makers, notably Microsoft, by using more free, open-source software.

cyber.law.harvard.edu/epolicy/

“It was the standardization around HTML that allowed the web to take off. It was not only the fact that it is standard, but the fact that its open and the fact that it is royalty-free.

So what we saw on top of the web was a huge diversity and different business which are built on top of the web given that it is an open platform.

If HTML had not been free, if it had been proprietary technology, then there would have been the business of actually selling HTML and the competing JTML, LTML, MTML products. Because we wouldn't have had the open platform, we would have had competition for these various different browser platforms, but we wouldn't have had the web. We wouldn't have had everything growing on top of it.

So I think it very important that as we move on to new spaces we must keep the same openness we that had before. We must keep an open internet platform, keep the standards for the presentation languages common and royalty free. So that means, yes, we need standards, because the money, the excitement is not competing over the technology at that level. The excitement is in the businesses and the applications that you built on top of the web platform.”

-- Tim Berners-Lee (W3C, inventor of the world wide web)

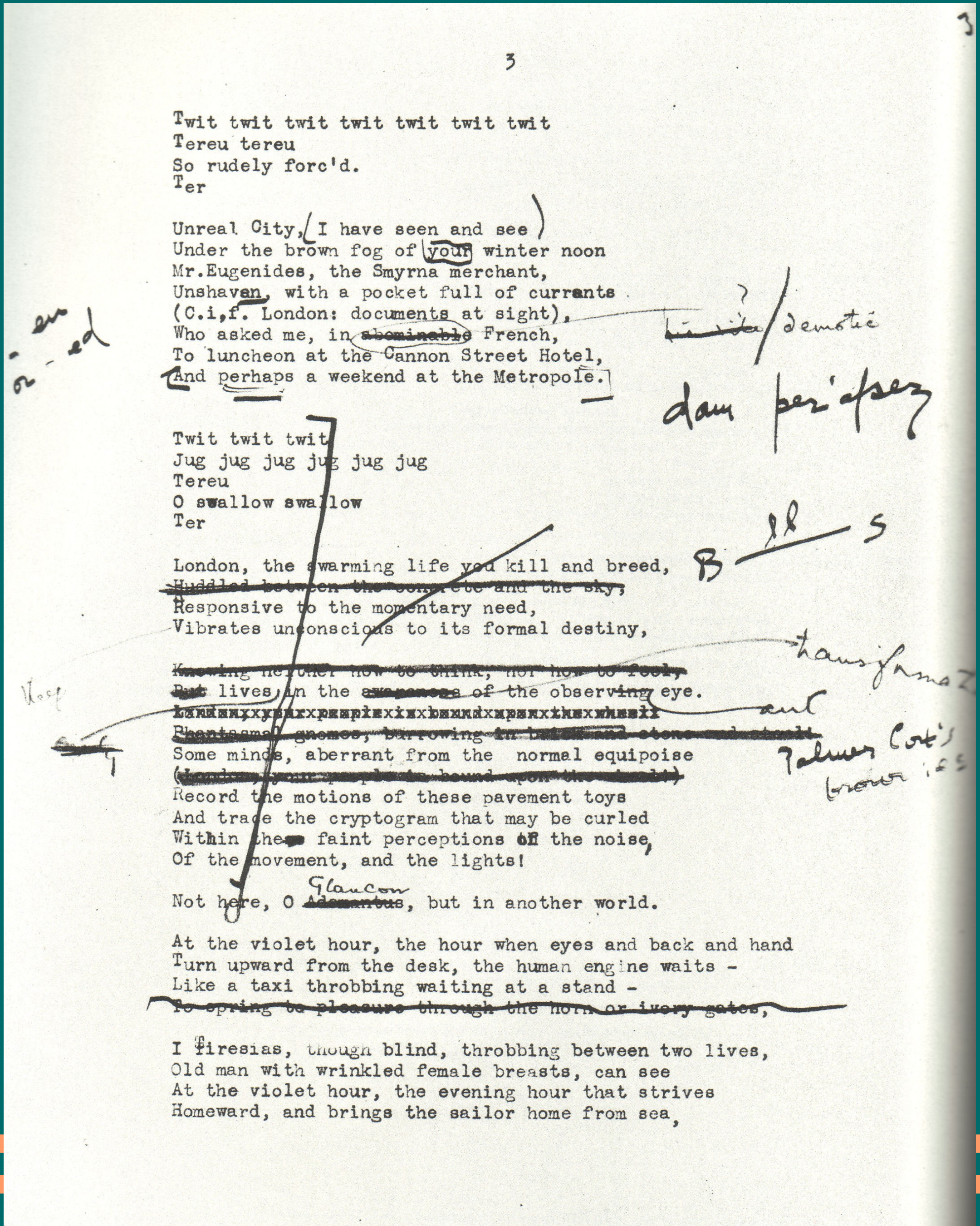
Why not just use PDF/A ?

- PDF is a good representation of the final, frozen, never-to-be-edited digital equivalent of the printed page.

- But you lose some things:
  - No spreadsheet formulas, so you can’t figure out where the numbers came from.
  - Review/comment threads are lost or collapsed, so the record of who changed what when is lost.
  - Mathematical equations are just images, diagrams are now just pictures, making then impossible for assistive technologies to render them properly to the blind

  - Best to capture the document in the fullest information state

Consider “The Wasteland”



3

Twit twit twit twit twit twit twit
Tereu tereu
So rudely forc'd.
Ter

Unreal City, (I have seen and see)
Under the brown fog of your winter noon
Mr.Eugenides, the Smyrna merchant,
Unshaven, with a pocket full of currants
(C.i.f. London: documents at sight),
Who asked me, in ~~abominable~~ French, demotic
To luncheon at the Cannon Street Hotel,
And perhaps a weekend at the Metropole.

-en
or -ed

dam per'apsez

Twit twit twit
Jug jug jug jug jug jug
Tereu
O swallow swallow
Ter

London, the swarming life you kill and breed, B ll S
~~Huddled between the concrete and the sky,~~
Responsive to the momentary need,
Vibrates unconscious to its formal destiny,

~~Knowing neither how to think, nor how to feel,~~ transforms?
~~But~~ lives in the ~~awareness~~ of the observing eye.
~~Londonxxyourxpeoplexisxboundxuponxthexwheelx~~ aut
~~Phantasmal gnomes, burrowing in brick and stone and steel!~~ Palmer Cox's brownies
Some minds, aberrant from the normal equipoise
~~(London, your people is bound upon the wheel!)~~
Record the motions of these pavement toys
And trace the cryptogram that may be curled
Within these faint perceptions of the noise,
Of the movement, and the lights!

Keep

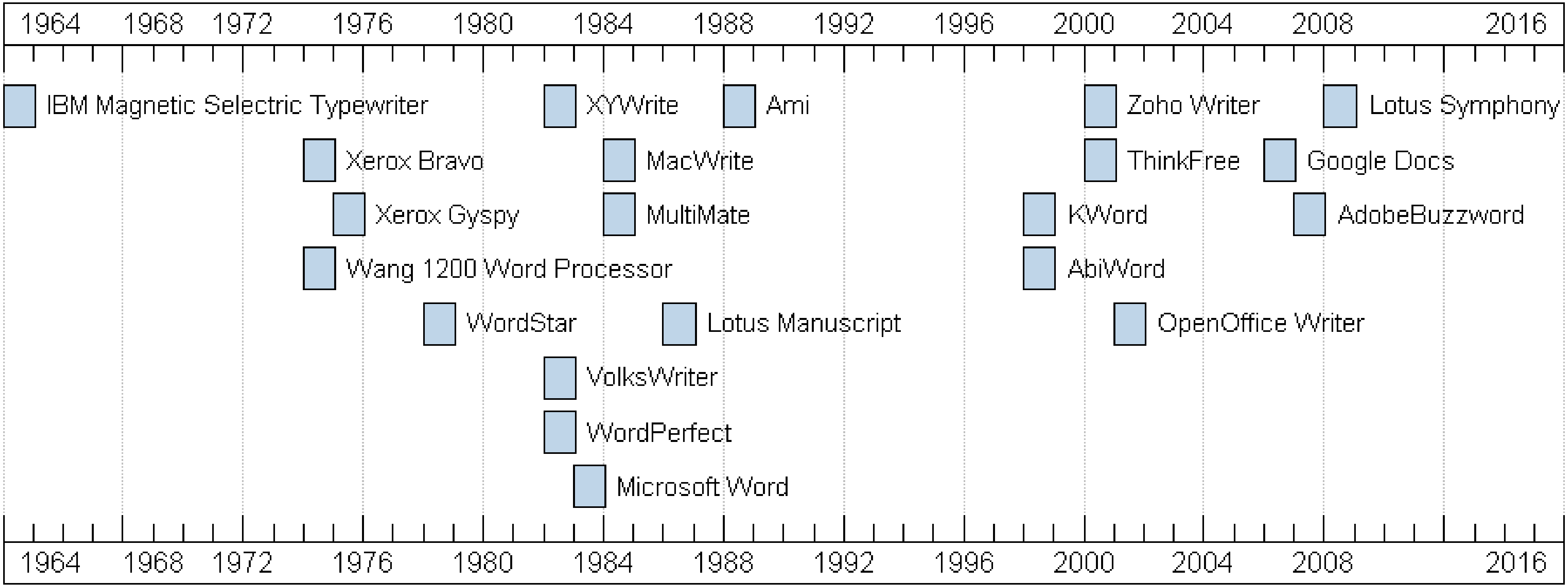
Not here, O ~~Adeimantus~~ Glaucon, but in another world.

At the violet hour, the hour when eyes and back and hand
Turn upward from the desk, the human engine waits -
Like a taxi throbbing waiting at a stand -
~~To spring to pleasure through the horn or ivory gates,~~

I Tiresias, though blind, throbbing between two lives,
Old man with wrinkled female breasts, can see
At the violet hour, the evening hour that strives
Homeward, and brings the sailor home from sea,

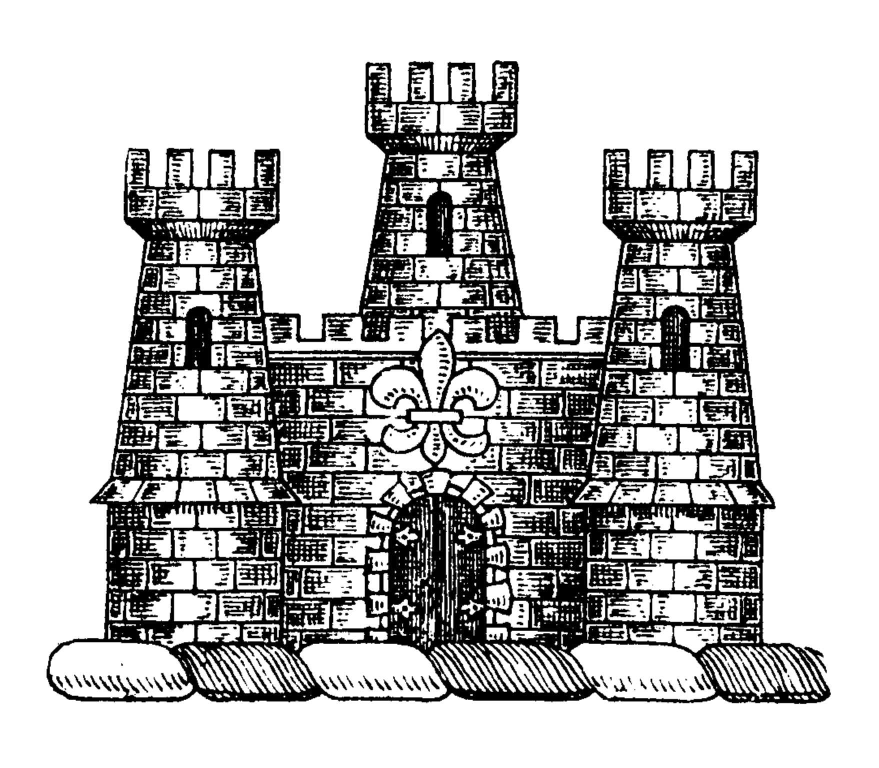
History of Document Formats

Word Processors



The age of proprietary formats

- Created by a single vendor
- Controlled a single vendor
- Evolved by a single vendor



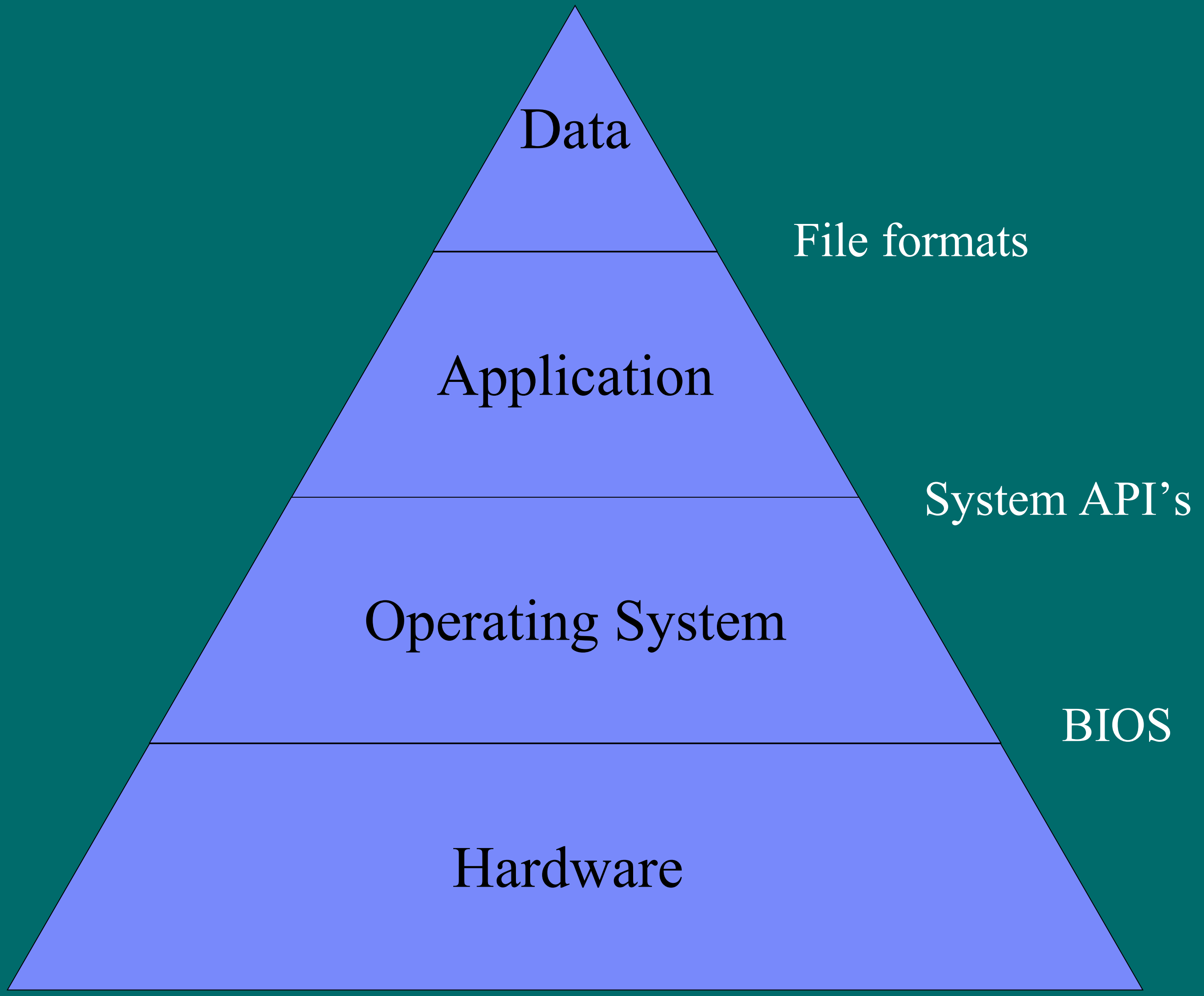
Restrictive Licensing

"...you may use documentation identified in the MSDN Library portion of the SOFTWARE PRODUCT as the file format specification for Microsoft Word, Microsoft Excel, Microsoft Access, and/or Microsoft PowerPoint ("File Format Documentation") solely in connection with your development of software product(s) that operate in conjunction with Windows or Windows NT that are not general purpose word processing, spreadsheet, or database management software products or an integrated work or product suite whose components include one or more general purpose word processing, spreadsheet, or database management software products."

MSDN Licence, 1998

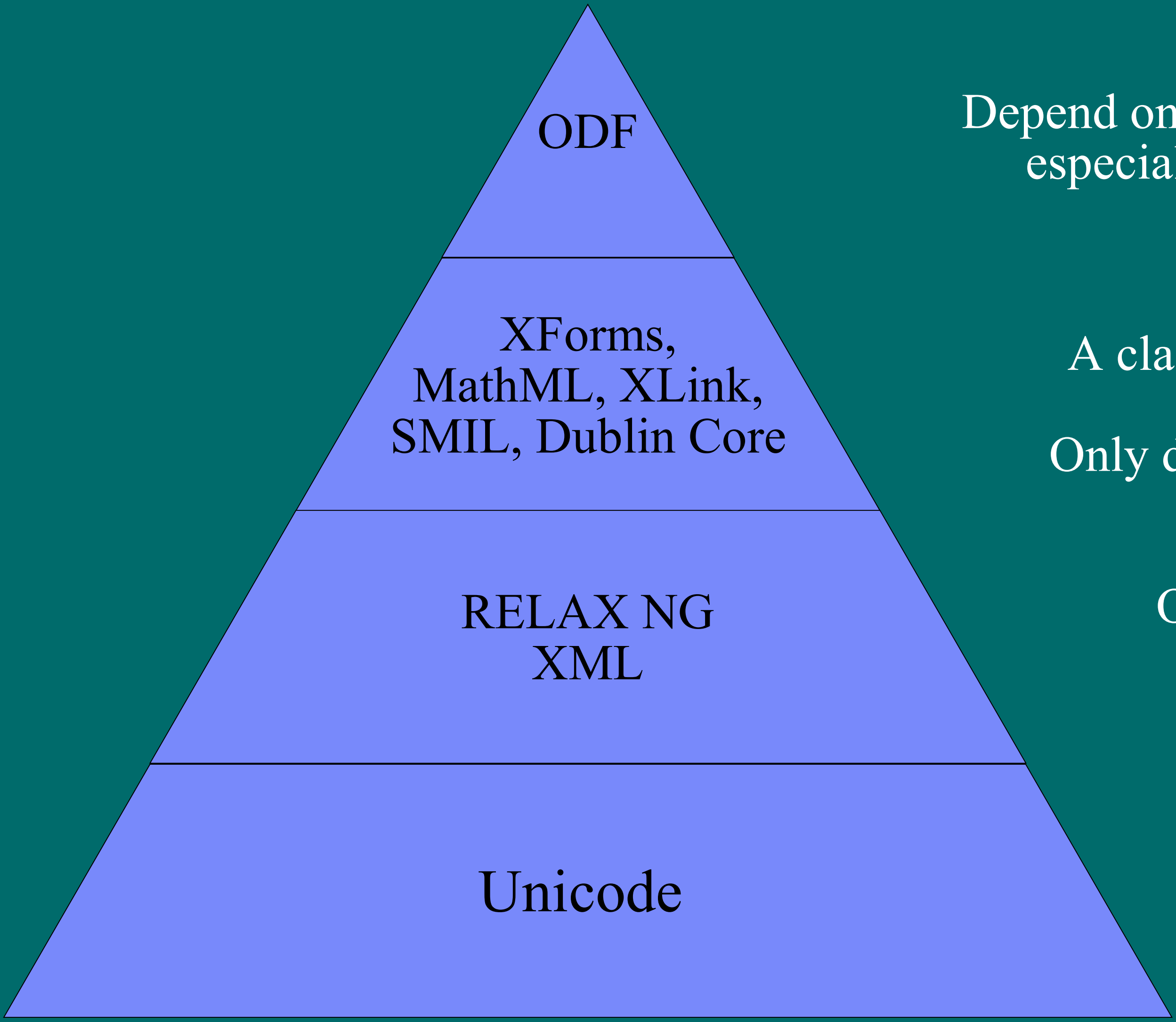
Traditional view of commercial software

Every layer depends on the layers beneath it



The rigidity of this model is being overcome

- There are storage devices that are independent of hardware platform
  - E.g., the various ISO optical disk standards
- There are applications which are independent of operating systems
  - OpenOffice, Firefox
- There are file formats which are independent of applications
  - SGML
  - XML
  - HTML
  - PNG
  - ODF



The ODF Way:

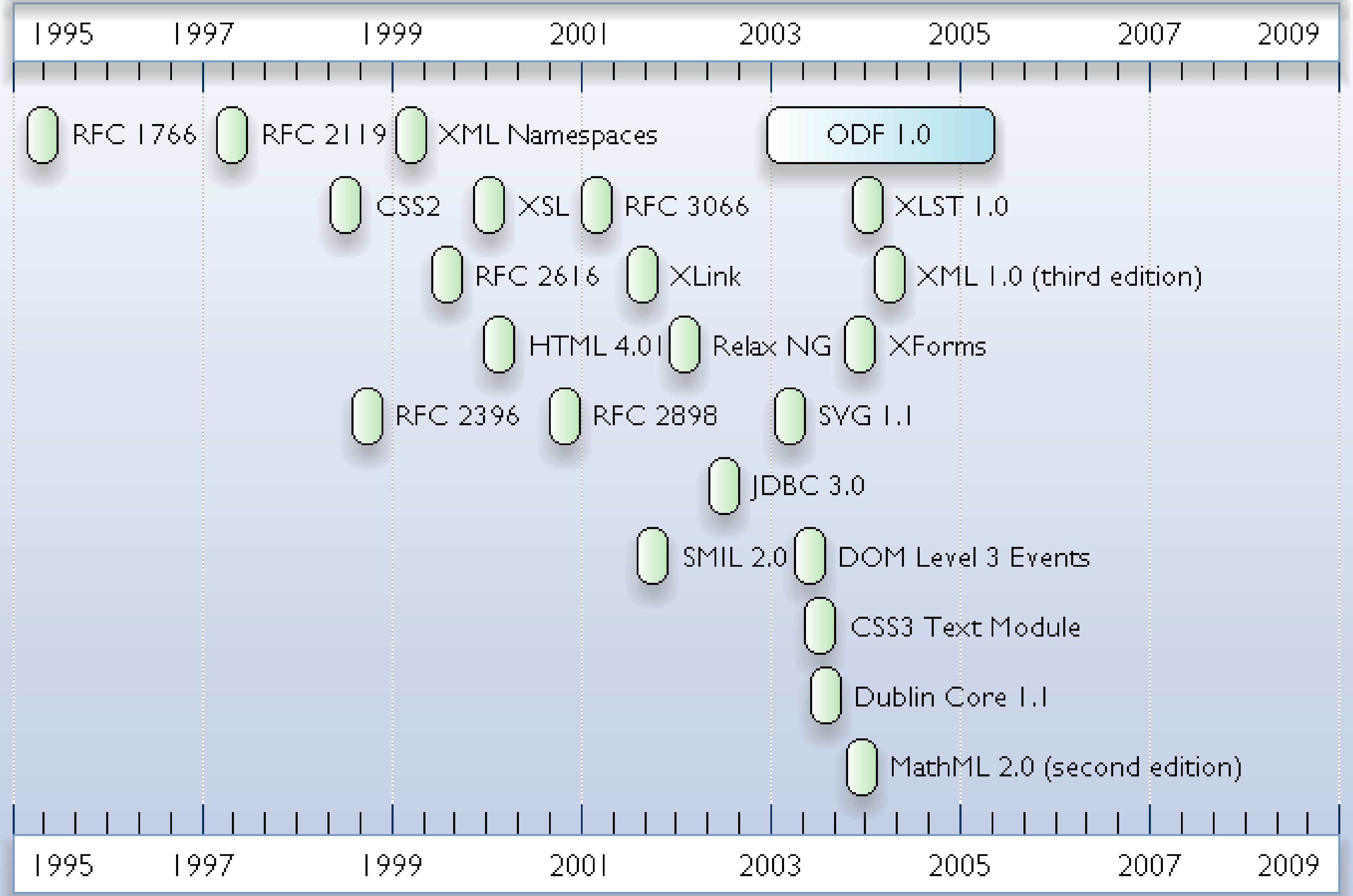
Depend on other pre-existing standards, especially bedrock web standards

A classic architectural principle says:

Only depend on things more stable than yourself.

ODF tries to follow that rule.

ODF Dependencies



Reuse of standards

“If I have seen a little further it is by standing on the shoulders of Giants.”

Isaac Newton, letter to Robert Hooke, 1676

Choose reuse because:

- Reduced time to write specification
- Higher quality specifications
- Can leverage existing community expertise
- Can leverage existing education materials
- Better interop, especially in a word of promiscuous mashups, not monolithic silos
- Network effects – synergy is good

OASIS and the ODF TC

OASIS

Advancing open standards for the information society

- Organization for the Advancement of Structured Information Standards
- Formally called “SGML Open”, Founded in 1993
- 600 corporation/organizations represented
- 5,000 participants from 100 countries
- 70 active technical committees (TC's)

- DITA, DocBook, ebXML, Election Markup Language, SAML, UBL, WebCGM

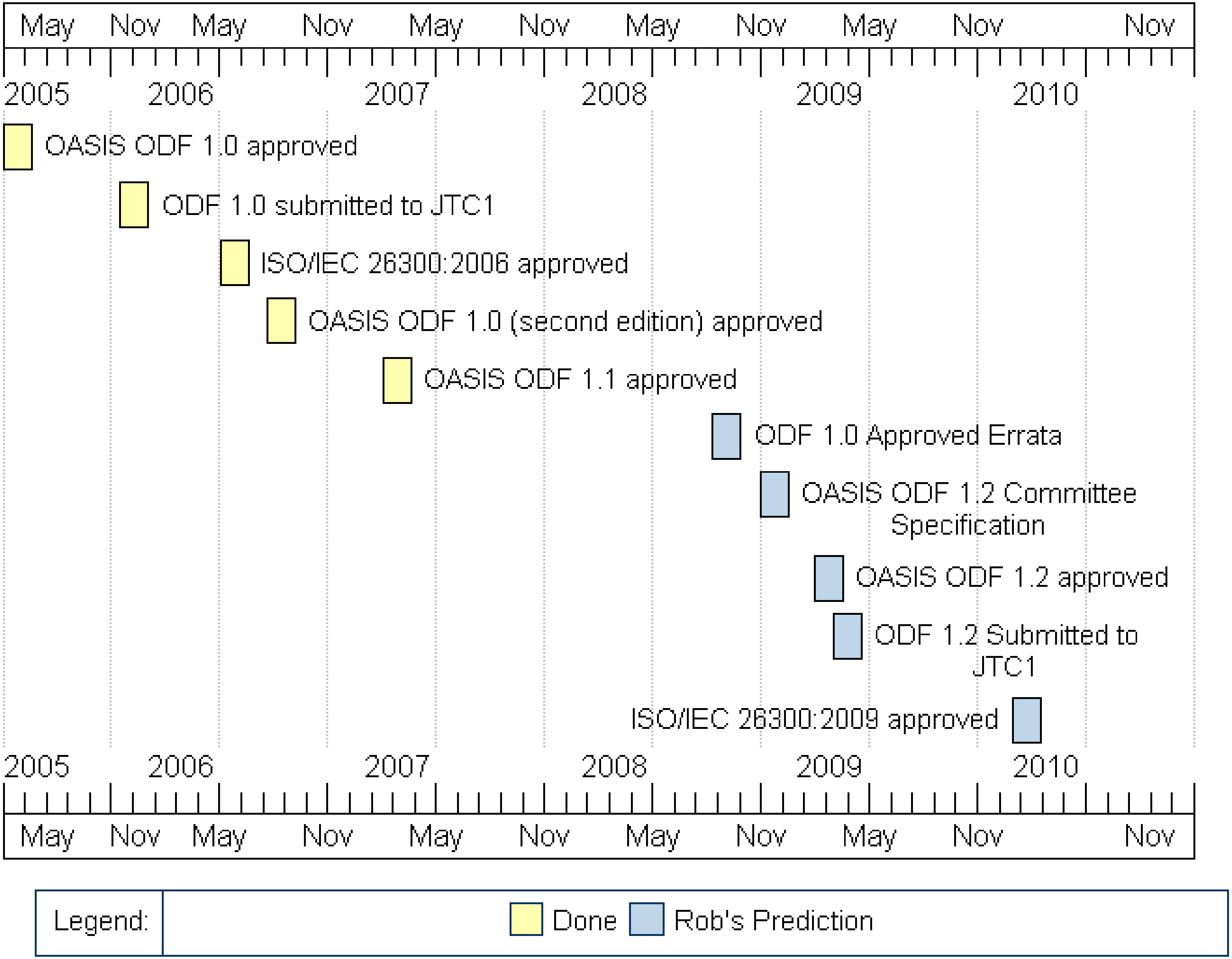
- http://www.oasis-open.org/

OASIS ODF TC

- ~12 regularly attending TC members
  - IBM, Sun, Novell, Microsoft, Google, KOffice, South African Dept. of Science & Technology and assorted individual contributors (consultants, academics, etc.)
  - Microsoft just joined us.

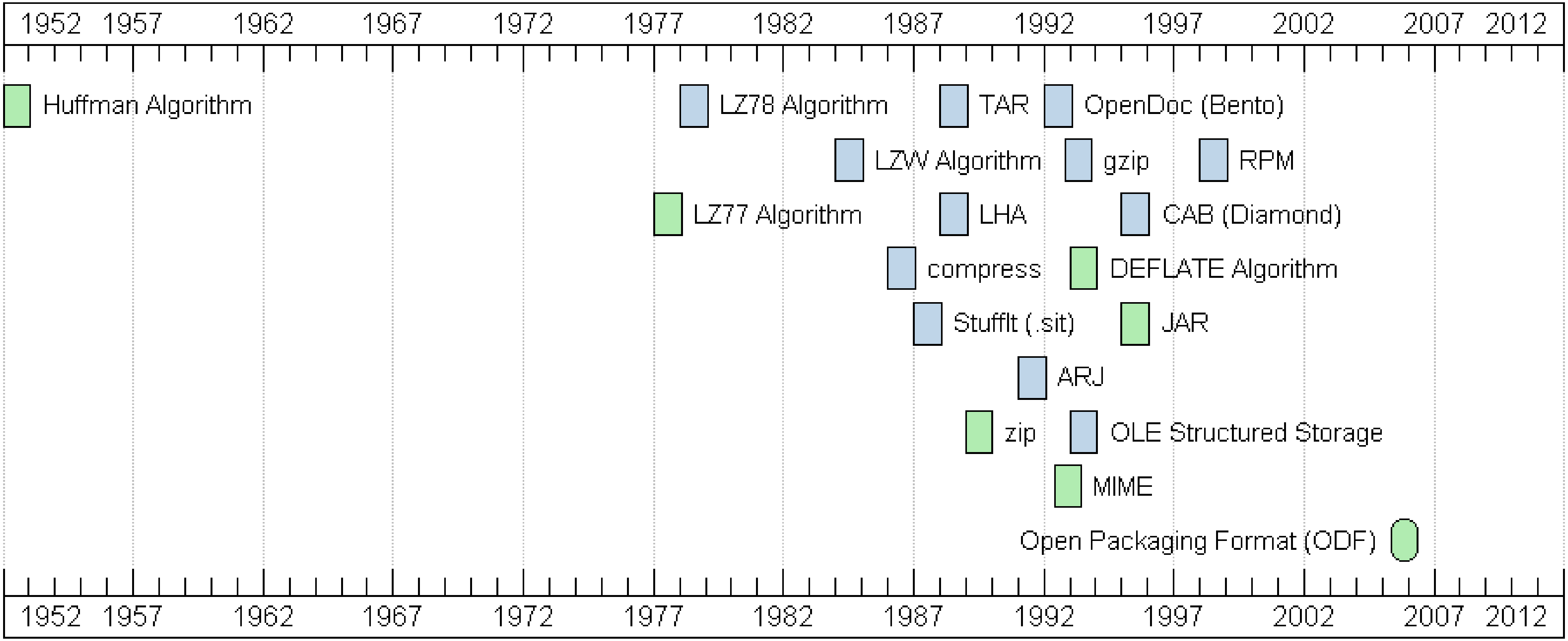
  - Three formal subcomittees that meet independently:
    - Accessibility
    - OpenFormula
    - Metadata

ODF Timeline



Packaging Model

History of Packaging



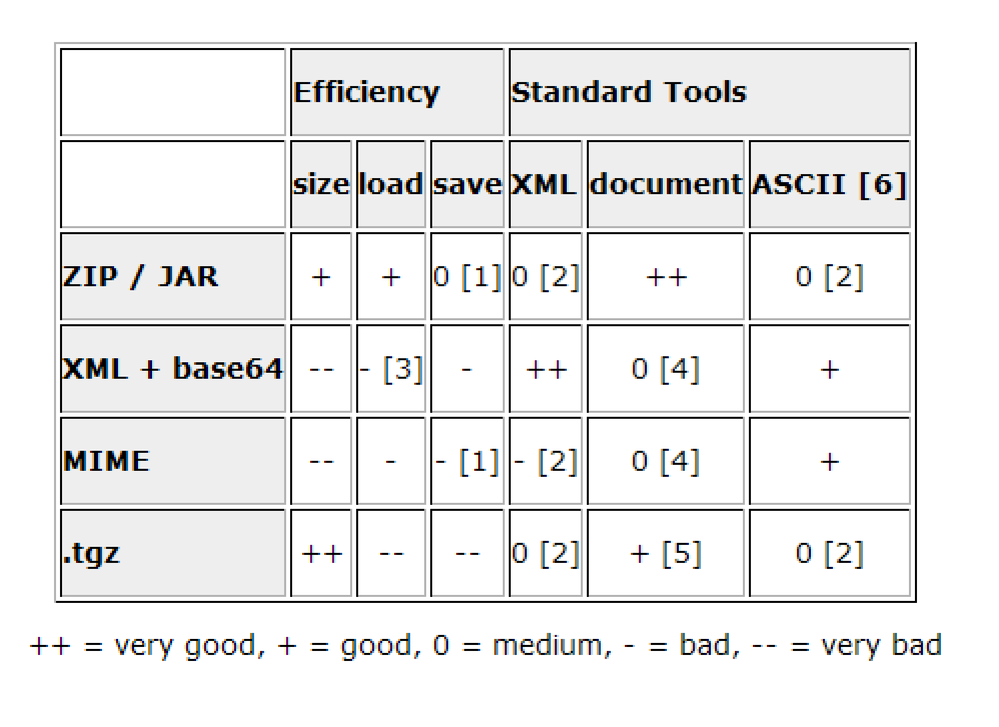
Design Requirements *

- Efficient Operation
  - Small file sizes
  - Ability to independently load and update subdocuments
- Compatibility with existing tools
  - Subdocuments processable with standard tools
  - The document itself should also
  - Should be text-based
- Security
  - Privacy
  - Integrity
- Extensibility

*based on OO report on their analysis: http://xml.openoffice.org/package.html

Options considered

- Zip/JAR
- XML with base64 embeddings
- MIME
- .tgz files



	Efficiency			Standard Tools		
	size	load	save	XML	document	ASCII [6]
ZIP / JAR	+	+	0 [1]	0 [2]	++	0 [2]
XML + base64	--	- [3]	-	++	0 [4]	+
MIME	--	-	- [1]	- [2]	0 [4]	+
.tgz	++	--	--	0 [2]	+ [5]	0 [2]

++ = very good, + = good, 0 = medium, - = bad, -- = very bad

So what do we have here?

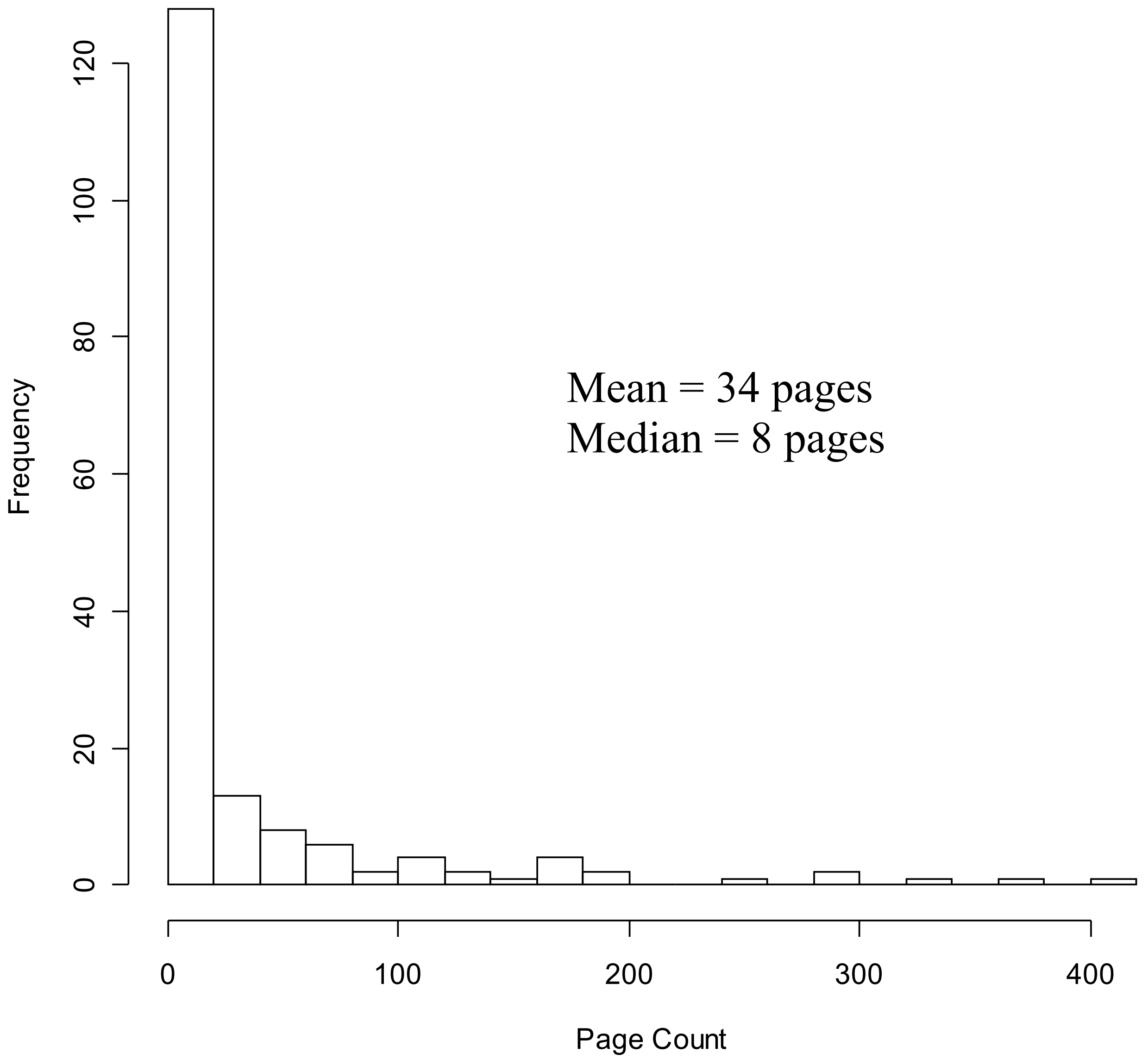
- A Zip file containing:
  - An XML manifest file
  - Additional XML files to describe the doc's content, styles and metadata
  - Possibly additional binary files for embedded media files (images, etc)
  - Same idea works for spreadsheet, text and presentation documents
- Quick demonstration

Packaging

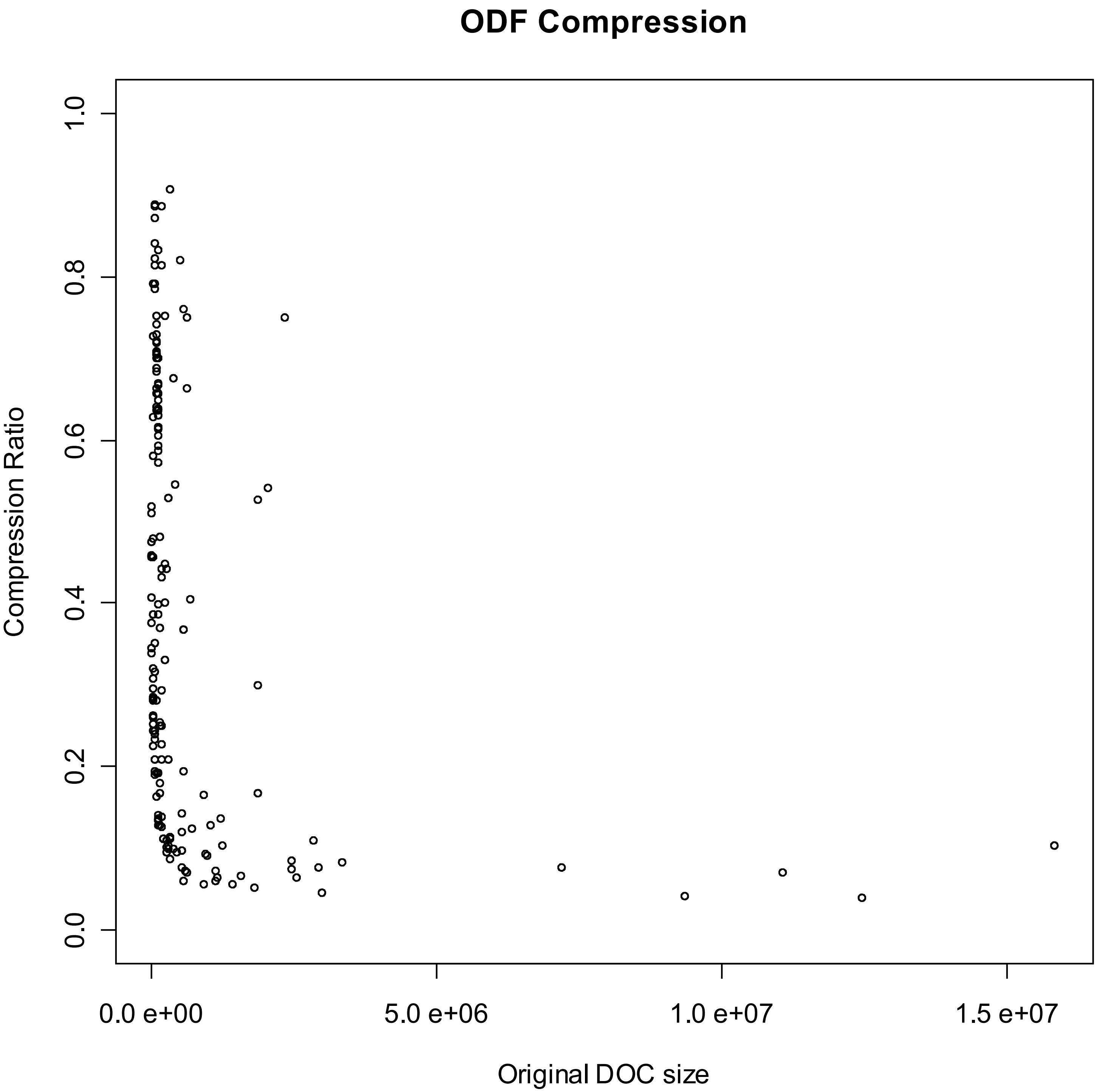
- Knowing just the packaging is enough to:
  - Discover, edit, remove or index metadata
  - Route documents
  - Apply and verify digital signatures
  - Replace bulk content
  - Search for viruses
  - Check security constraints

Some comparative metrics

- 176 Word documents from a document library
- Convert all to ODF format
- Record:
  - Number of pages
  - ZIP size
  - Numbered of contained files
  - Numbered of contained XML files
  - Total uncompressed size of contained files
  - Total uncompressed size of contained XML files.



* All charts and calculations done with the excellent open source “R” environment; http://www.r-project.org/



Compression ratio =
Size of ODF/Size of DOC

Mean = 0.38

Internationalization

- Character set
  - Based on XML 1.0 which supports Unicode 3.1
  - Supports most writing systems of the world
- Bidirection Text
  - Fully supported
- IRI’s (Unicode URL’s)
  - Fully supported

- Implementations of ODF, such as OpenOffice.org are along the most translated, most internationalized software applications in the world

Accessibility

ODF Accessibility – Initial Problem

- In US Federal Government bids software must be accessible by Persons with Disabilities.
  - Section 508 – US Rehabilitation Act
- Now becoming important for States, e.g. Commonwealth of Massachusetts
  - Also California, Texas, Minnesota
- ODF Accessibility issues were raised by lobbyists in Massachusetts.
- Leaders from IBM worked with the ODF TC to form the Accessibility SC (see next slide).
- OASIS ODF AccSC was formed and responded very quickly.

Committee Formed

- IBM
- Sun
- Design Science
- The Paciello Group
- Capital Accessibility
- Institute of Community Inclusion
- Royal National Institute for the Blind
- Duxbury Systems

GAP Analysis

- The Accessibility Subcommittee (AccSC) was formed in January 26, 2006.
- A GAP analysis was conducted.
- Comparison to W3C WCAG 1.0 and the Microsoft Office suite
- Nine issues were identified and submitted to the TC during May 2006.

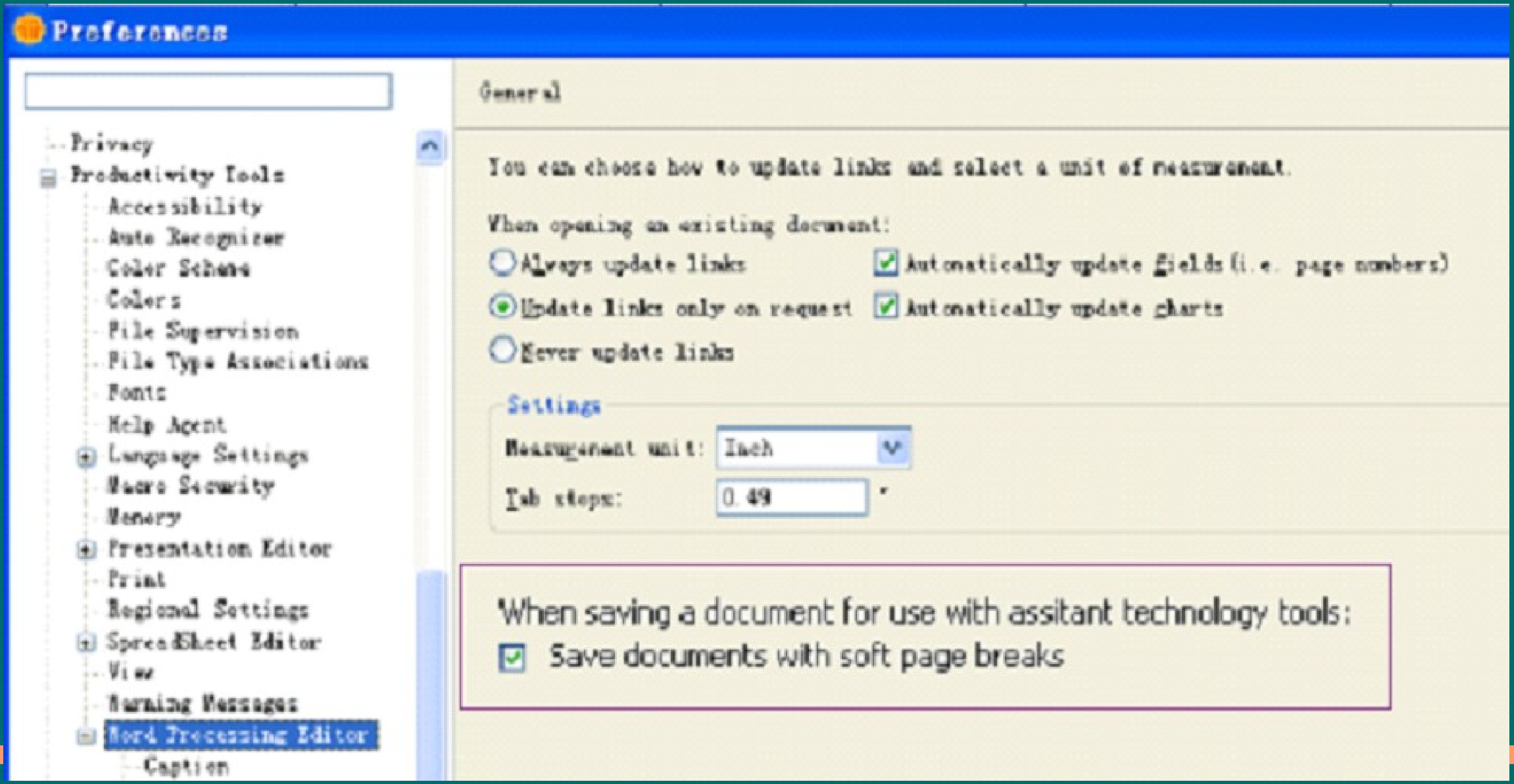
TC Approval

- Eight of the nine issues were approved
- Tables as first class presentation objects will be in ODF 1.2
  - 1.1 workaround – embed as spreadsheet
- ODF 1.1 announced February 13, 2007

9 Fixes (as seen in Symphony)

1. Soft page breaks

- Notes 8 uses <text:soft-page-break> element and <text:use-soft-page-breaks> attribute



General

You can choose how to update links and select a unit of measurement.

When opening an existing document:

○ Always update links
● Update links only on request
○ Never update links

☑ Automatically update fields (i.e. page numbers)
☑ Automatically update charts

Settings

Measurement unit: Inch

Tab stops: 0.49 "

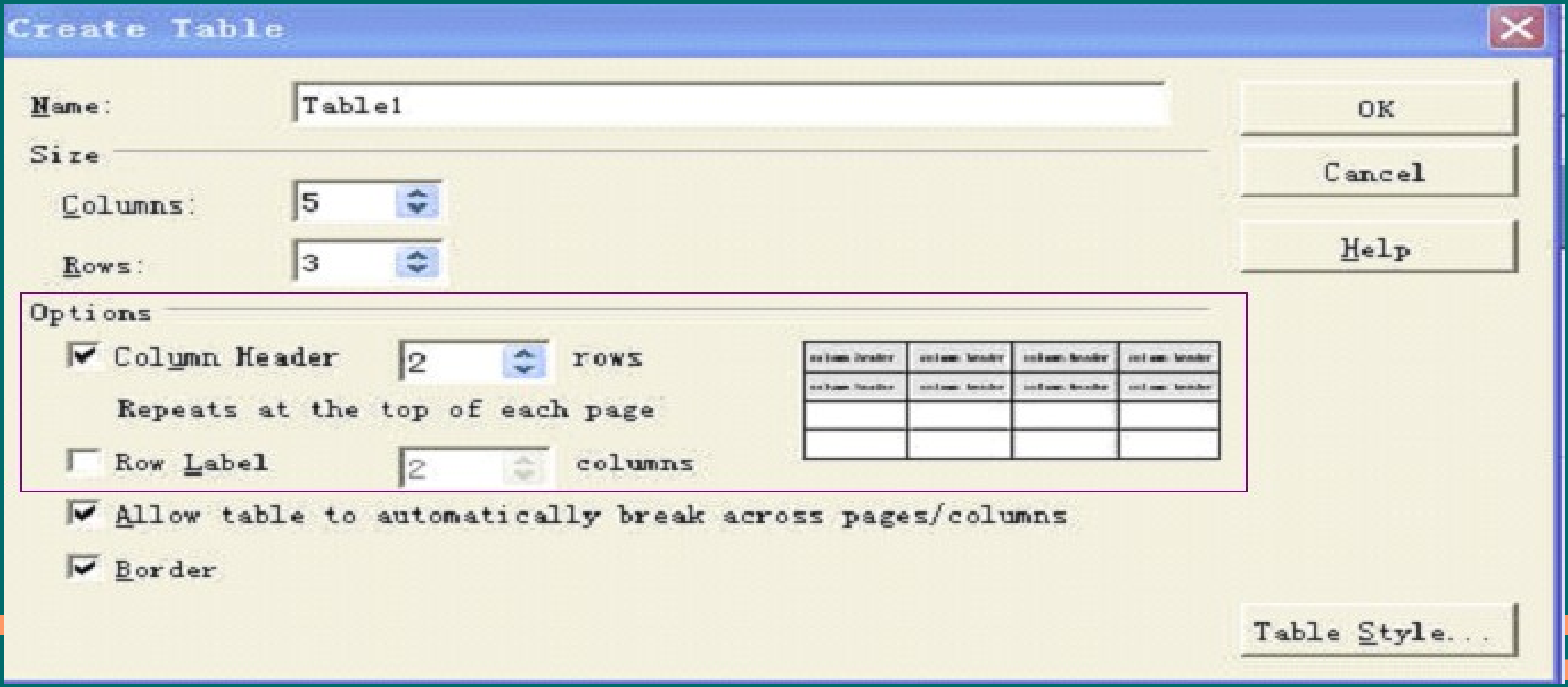
When saving a document for use with assitant technology tools:

☑ Save documents with soft page breaks

9 Fixes (as seen in Symphony)

2. Support table header structural markup

- Notes 8 supports table row/column headers.
- IAccessibleTable:RowHeader and IAccessibleTable::ColumnHeader provide accessible information



Create Table

Name: Table1

Size

Columns: 5

Rows: 3

Options

☑ Column Header 2 rows

Repeats at the top of each page

□ Row Label 2 columns

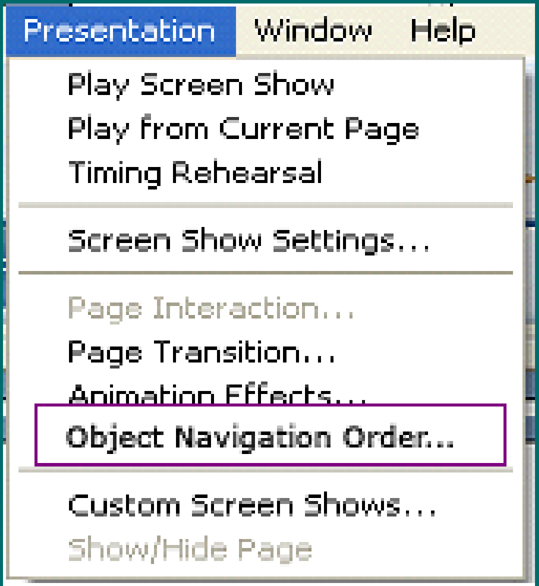
☑ Allow table to automatically break across pages/columns

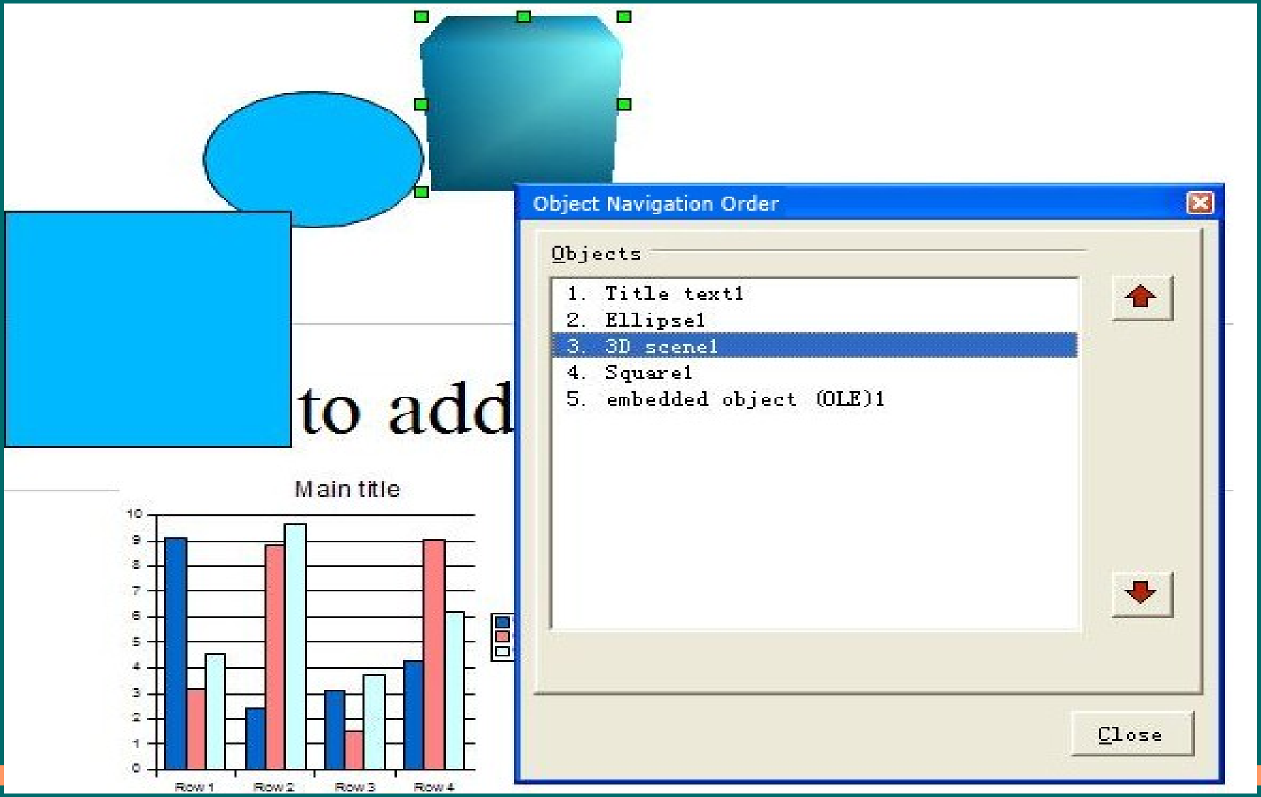
☑ Border

9 Fixes (as seen in Symphony)

3. Provide for author specified, logical navigation in presentations

- Special dialog for logical navigation order.





9 Fixes (as seen in Symphony)

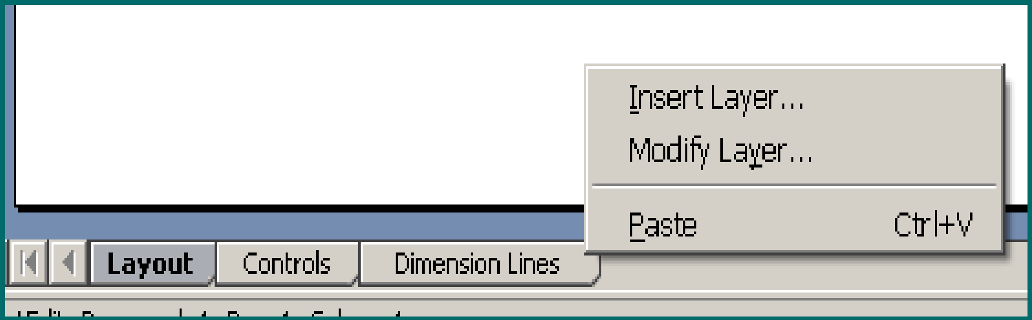
4. Alternative text for image map elements

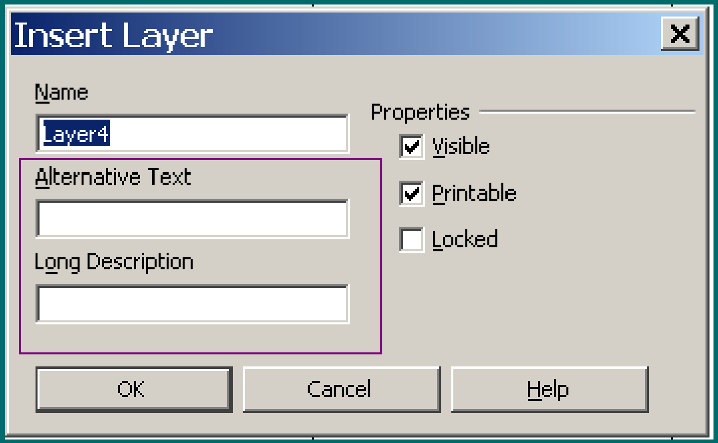
- Image map elements have <svg:title> (alt text) and <svg:desc> (description).
- IAccessibleHyperlink::description will display <svg:title>. If <svg:title> is null IAccessibleHyperlink::description will display the <svg:desc>.

9 Fixes (as seen in Symphony)

5. Alternative Text for Drawing Layer

Alt text and long description fields added.



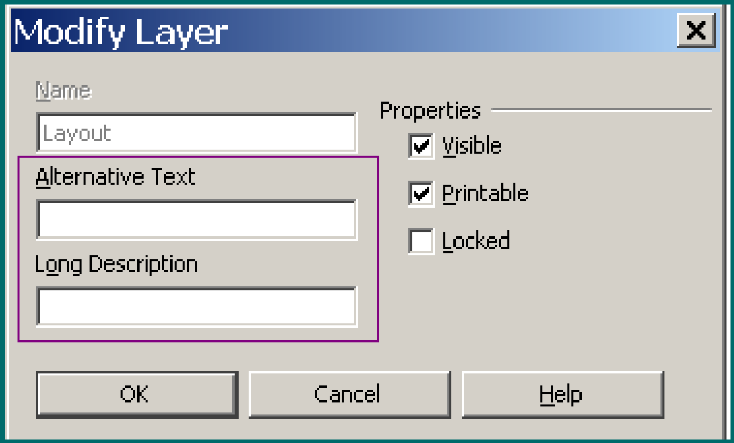


Name: Layer4

Alternative Text:

Long Description:

Properties
- ☑ Visible
- ☑ Printable
- ☐ Locked



Name: Layout

Alternative Text:

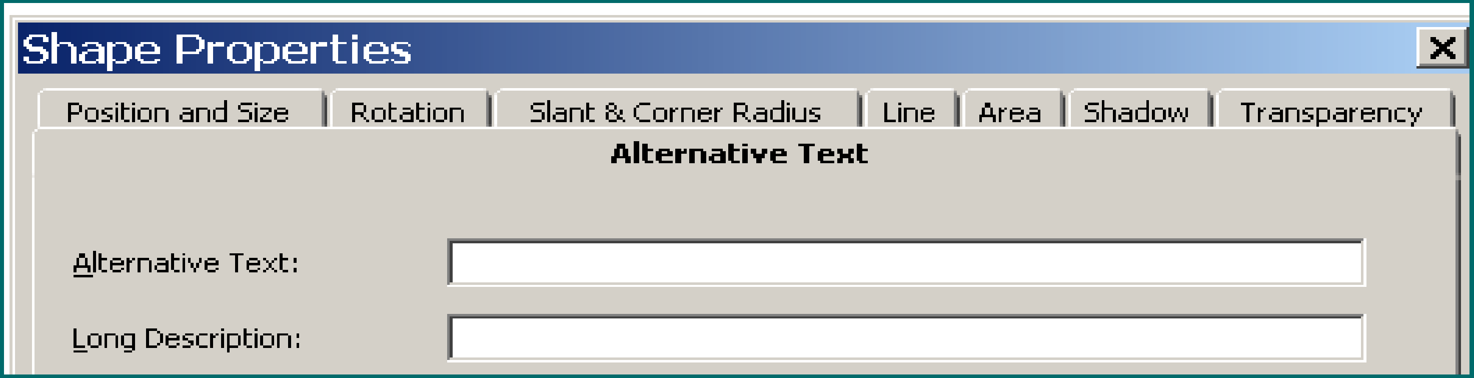
Long Description:

Properties
- ☑ Visible
- ☑ Printable
- ☐ Locked

9 Fixes (as seen in Symphony)

6. Alternative Text for Drawing Objects

- Text fields for alternative text & long description.
- Exposed via MSAA's name and description.



Shape Properties

Alternative Text

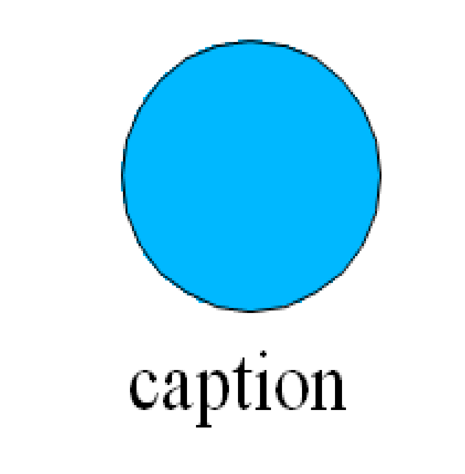
Alternative Text:

Long Description:

9 Issues - Notes 8 Implementation

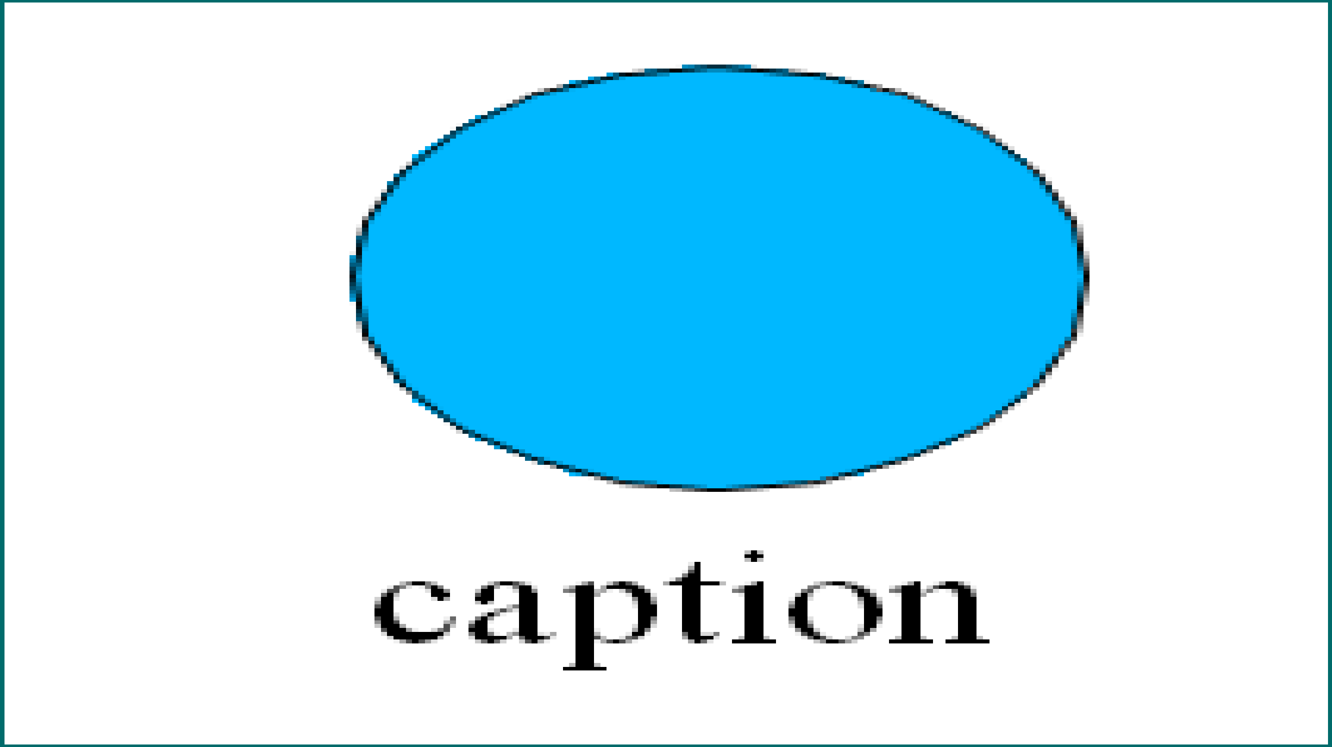
7. Relations between Objects & Captions

- Relation is exposed via ODF <draw:caption-id>
- And via the IAccessible2 describedBy relation.



9 Fixes (as seen in Symphony)

8. Establish text hints for hyperlinks



The description is saved to <svg:title> and is exposed by IAccessibleHyperlink:description.

9 Fixes (as seen in Symphony)

9. Tables in Presentations

- Work-around for ODF 1.1
  - Encoded as embedded spreadsheet
- To be encoded as table:table in ODF 1.2

Fostering Innovation - DAISY

- ODF 1.1 based docs can be used as content for DAISY talking books.
- Dave Pawson from RNIB is an XML and XSLT expert and he made very significant contributions to the spec.
- Soft page breaks were added to ODF 1.1
- ODF 1.1 content is being transformed into DAISY content.
- Open standards developed by industry experts facilitate innovation.

Fostering Innovation - Duxbury

- Duxbury Systems
- Tooling to create Braille content
- Compared to Word 2003 Duxbury gives ODF higher marks in the area of documentation, simplicity -- and reusability:
  - ODF importer shares code in common with importers for OpenXML, HTML, DAISY/NIMAS, and generic "XML".
  - ODF allowed re-use of table importer between word processing and spreadsheet editors
  - This ease of reuse lowers the price barrier for creating Braille content.

Duxbury - ODF to Braille

- Unpack content.xml
- Preprocess
  - Resolve issues that cause XML parsing problems
- Transform with XSLT (od.xsl)
  - “Pick out” the parts of interest
  - Omit extraneous items
  - Set up “hints” for post-processing
- Map styles in XML (xsmod.xml)
  - Allows application to different document standards
- Postprocess
  - Re-encode characters
  - Eliminate most empty nodes
  - Transform “hints” to direct DBT codes
- Pack <filename>.dxp

Metadata

Metadata

- ODF includes bibliographic metadata according to Dublin Core Metadata:
  - title, description, subject, creator, date, language
- Also, includes additional metadata in ODF namespace:
  - Generator, keywords, initial-creator, printed-by, creation-date, print-date, template, editing-cycles, editing-duration, document-statistics, user-defined

- An implementation can also freely add their own metadata, either in an additional XML file or in an existing XML file.

- A finer-grained approach to content-level metadata will be added in ODF 1.2, based on the W3C’s RDF standard.

Extensibility

- Adding additional files to the document archive
  - Free to do this so long as you register the additional items in the manifest file
  - Good for adding any additional XML that describes or augments the entire document
- Adding additional XML markup to the content.xml in your own XML namespace

Toolkits & Libraries

The potential

- ODF – a platform and application neutral office file format
- Document data is no longer trapped in proprietary black box binaries

- This can lead to a “golden age” of document processing, both client and server side, with much innovation

- “We have it in our power to create the world over again” -- Thomas Paine

More than just editors

(20 Prototypical App Dev Scenarios)

1. Interactive creation in an a heavy-weight client application
2. Interactive creation in a light-weight web-based application
3. Collaborative (multi-author) editing
4. Automatic creation in response to a database query (report generation)
5. Indexing/scanning of document for search

20 Prototypical App Dev Scenarios

6. Scanning by anti-virus
7. Other types of scanning, perhaps for regulatory compliance, legal or forensic purposes
8. Validation of document, to specifications, house style guidelines, accessibility best practices, etc.
9. Read-only display of document on machine without the full editor (viewer)
10. Conversion of document from one editable format to another

20 Prototypical App Dev Scenarios

11. Conversion of document into a presentation format, such as PDF, PS, print or fax
12. Rendering of document via other modes such as sound or video (DAISY Talking Book)
13. Reduction/simplification of document to render on a sub-desktop device such as cell phone or PDA.
14. Import of data from an office document into a non-office application, i.e., import of spreadsheet data into statistical analysis software.
15. Export of data from a non-office application into an office format, such as an export of a spreadsheet from a personal finance application.

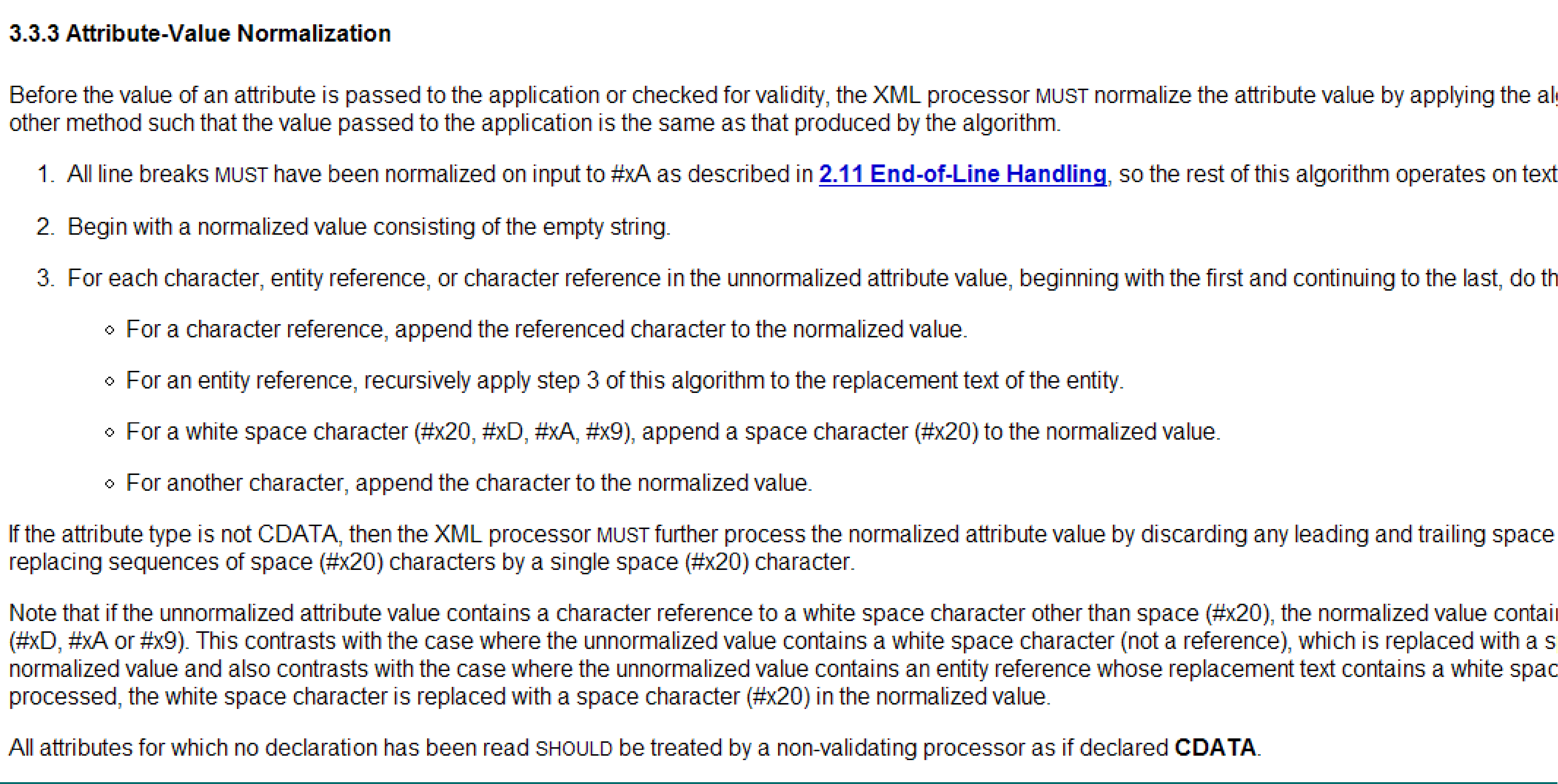
20 Prototypical App Dev Scenarios

16. Application which takes an existing document and outputs a modified version of that presentation, e.g., fills out a template, translates the language, etc.
17. Software which adds or verifies digital signatures on a document in order to control access (DRM)
18. Software which uses documents in part of a workflow, but treats the document as a black box, or perhaps is aware of only basic metadata.
19. Software which treats documents as part of a workflow, but is able to introspect the document and make decisions based on the content.
20. Software which packs/unpacks a document into relational database form.

The Problem

- 706 page ODF Specification
- No objections to it as a specification – it is what it needs to be
- Written from the perspective of word processor implementors
- Too much to ask the average app developer to master

Analogy with XML --
Who actually reads this stuff?



3.3.3 Attribute-Value Normalization

Before the value of an attribute is passed to the application or checked for validity, the XML processor MUST normalize the attribute value by applying the al
other method such that the value passed to the application is the same as that produced by the algorithm.

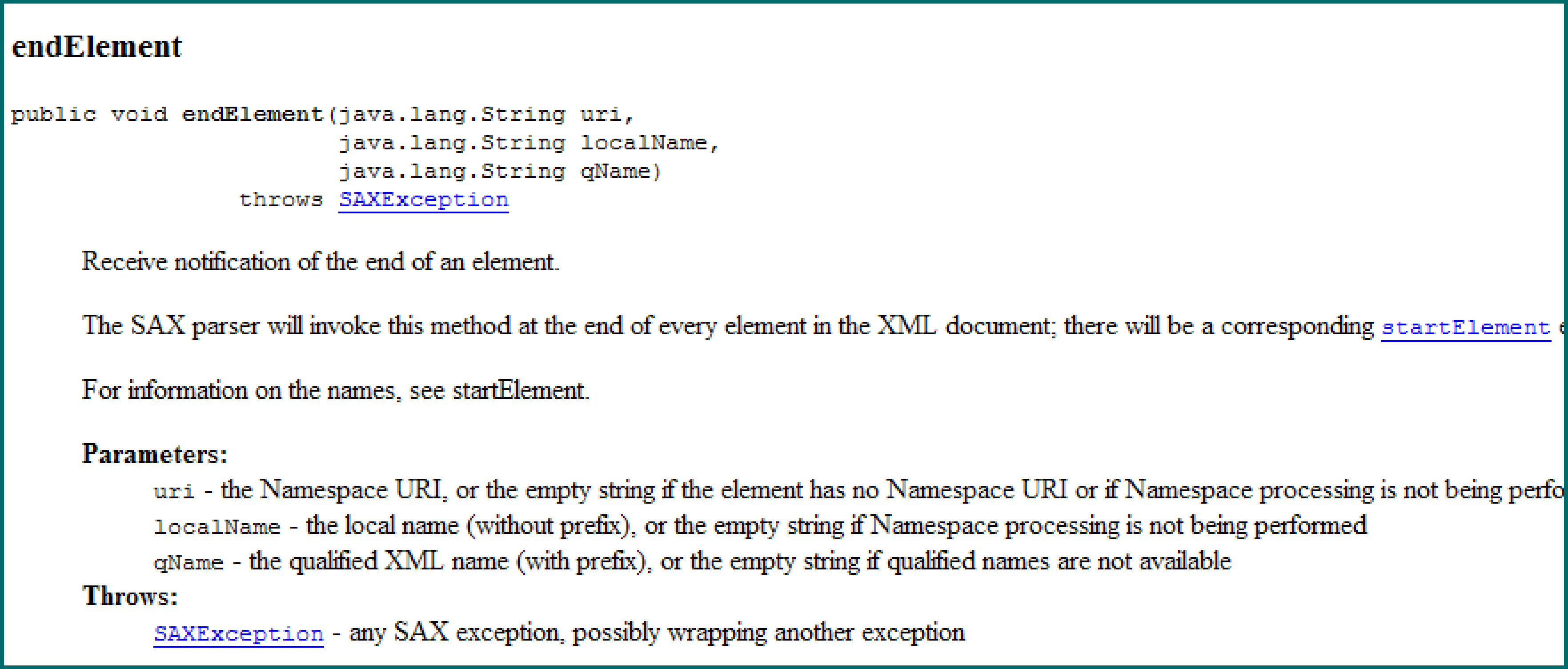
1. All line breaks MUST have been normalized on input to #xA as described in **2.11 End-of-Line Handling**, so the rest of this algorithm operates on text
2. Begin with a normalized value consisting of the empty string.
3. For each character, entity reference, or character reference in the unnormalized attribute value, beginning with the first and continuing to the last, do th
    - For a character reference, append the referenced character to the normalized value.
    - For an entity reference, recursively apply step 3 of this algorithm to the replacement text of the entity.
    - For a white space character (#x20, #xD, #xA, #x9), append a space character (#x20) to the normalized value.
    - For another character, append the character to the normalized value.

If the attribute type is not CDATA, then the XML processor MUST further process the normalized attribute value by discarding any leading and trailing space
replacing sequences of space (#x20) characters by a single space (#x20) character.

Note that if the unnormalized attribute value contains a character reference to a white space character other than space (#x20), the normalized value contai
(#xD, #xA or #x9). This contrasts with the case where the unnormalized value contains a white space character (not a reference), which is replaced with a s
normalized value and also contrasts with the case where the unnormalized value contains an entity reference whose replacement text contains a white spac
processed, the white space character is replaced with a space character (#x20) in the normalized value.

All attributes for which no declaration has been read SHOULD be treated by a non-validating processor as if declared **CDATA**.

What is really used is SAX



endElement

```
public void endElement(java.lang.String uri,
                       java.lang.String localName,
                       java.lang.String qName)
                throws SAXException
```

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding `startElement` e

For information on the names, see startElement.

Parameters:

`uri` - the Namespace URI, or the empty string if the element has no Namespace URI or if Namespace processing is not being perfo

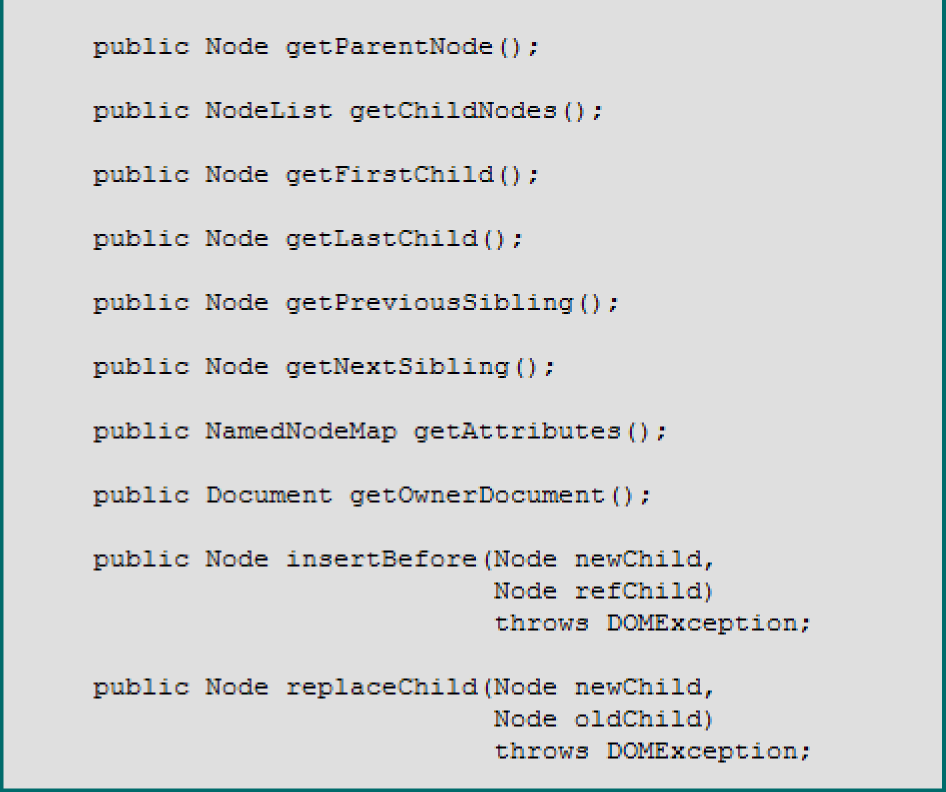
`localName` - the local name (without prefix), or the empty string if Namespace processing is not being performed

`qName` - the qualified XML name (with prefix), or the empty string if qualified names are not available

Throws:

`SAXException` - any SAX exception, possibly wrapping another exception

And DOM



```
public Node getParentNode();

public NodeList getChildNodes();

public Node getFirstChild();

public Node getLastChild();

public Node getPreviousSibling();

public Node getNextSibling();

public NamedNodeMap getAttributes();

public Document getOwnerDocument();

public Node insertBefore(Node newChild,
                         Node refChild)
                         throws DOMException;

public Node replaceChild(Node newChild,
                         Node oldChild)
                         throws DOMException;
```

The Challenge

We need an ODF API that exposes a higher level abstraction of ODF to application developers, so they can quickly become productive with ODF processing without having to master a 700 page specification

“Create a loan amortization spreadsheet in 30 lines of code”

Desirables

- Open source
- A convergent effort – bring together the projects that are already working in this area
- Wide range of language bindings, Java, Python, Ruby, C++, etc.
- Consider the API itself for standardization

This becomes the preferred way of working with ODF, the layer that the innovation builds upon

Some design ideas

- Useful to think of the toolkit in three classes:
  - The document representation – ODF DOM
    - Represents the state of the document, with get/set methods for manipulation. sheet.setCell("A1","hello")
  - A Parser class that takes an input stream and produces an ODF DOM object from it
  - A Serializer class that takes an ODF DOM object and writes it to an output stream

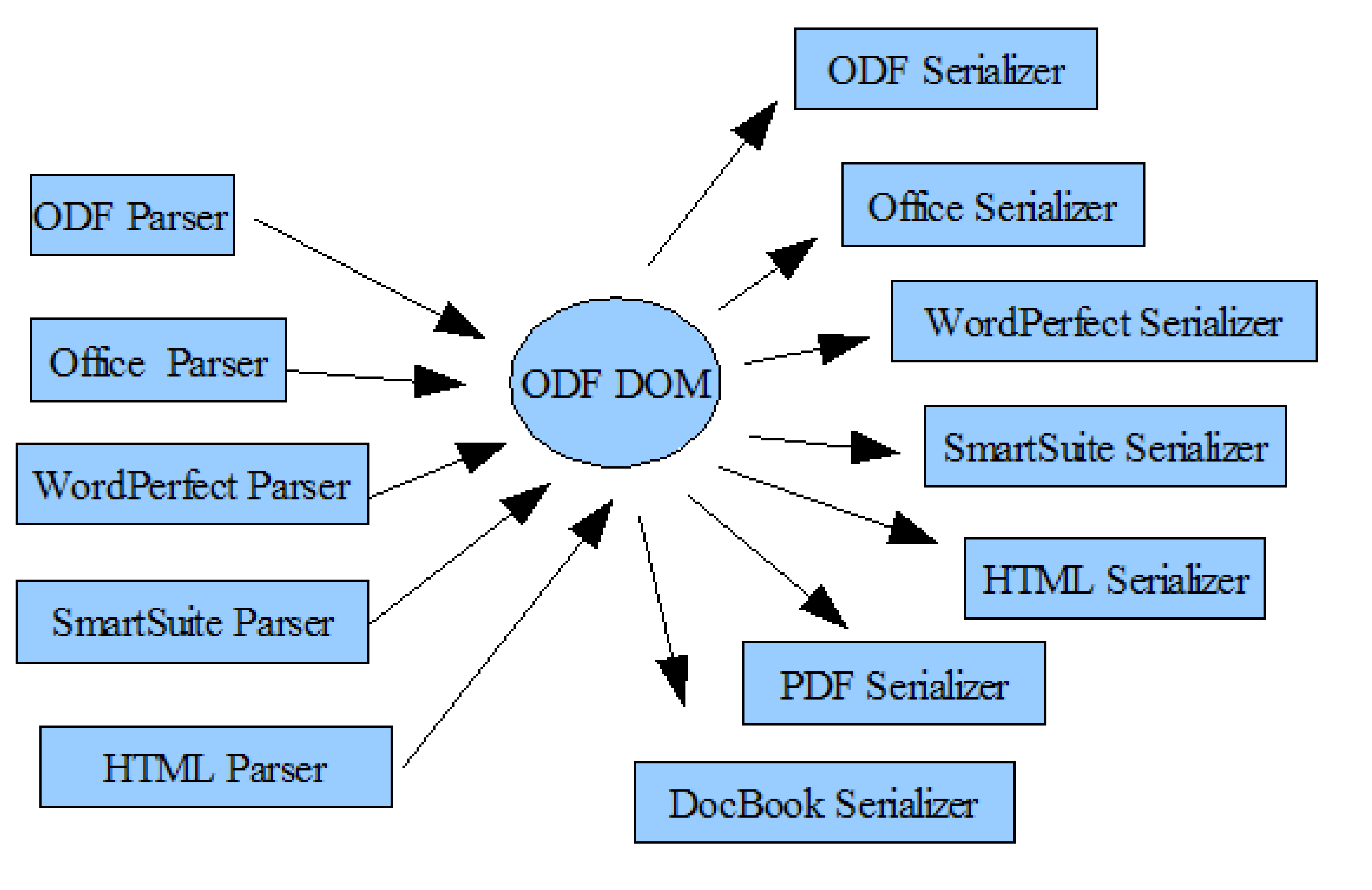
Modes of use

- Report generation
  - Create empty ODF DOM object, query a database, set data into the ODF object, then create Serializer to write it out to ODF document.
- Search engines
  - Create Parser, pass in stream to ODF document, create ODF DOM object, call methods to query document contents
- Mail Merge
  - Create parser, pass in stream, get ODF DOM object, find and replace content in the DOM, and then create a Serializer to write it out again

Key insight

- Factored this way, an additional opportunity emerges:
  - Is ODF the only source/destination format of the Parsers and Serializers? So long as they produce/consume ODF, who cares what the underlying data stream is?
  - Why not have an ExcelParser that reads an Excel document and creates an ODF DOM from it?
  - Why not have an PDFSeralizer that takes an ODF DOM document and renders it as PDF?

Hub and Spokes Model


ODF Parser

Office Parser

WordPerfect Parser

SmartSuite Parser

HTML Parser

ODF DOM

ODF Serializer

Office Serializer

WordPerfect Serializer

SmartSuite Serializer

HTML Serializer

PDF Serializer

DocBook Serializer

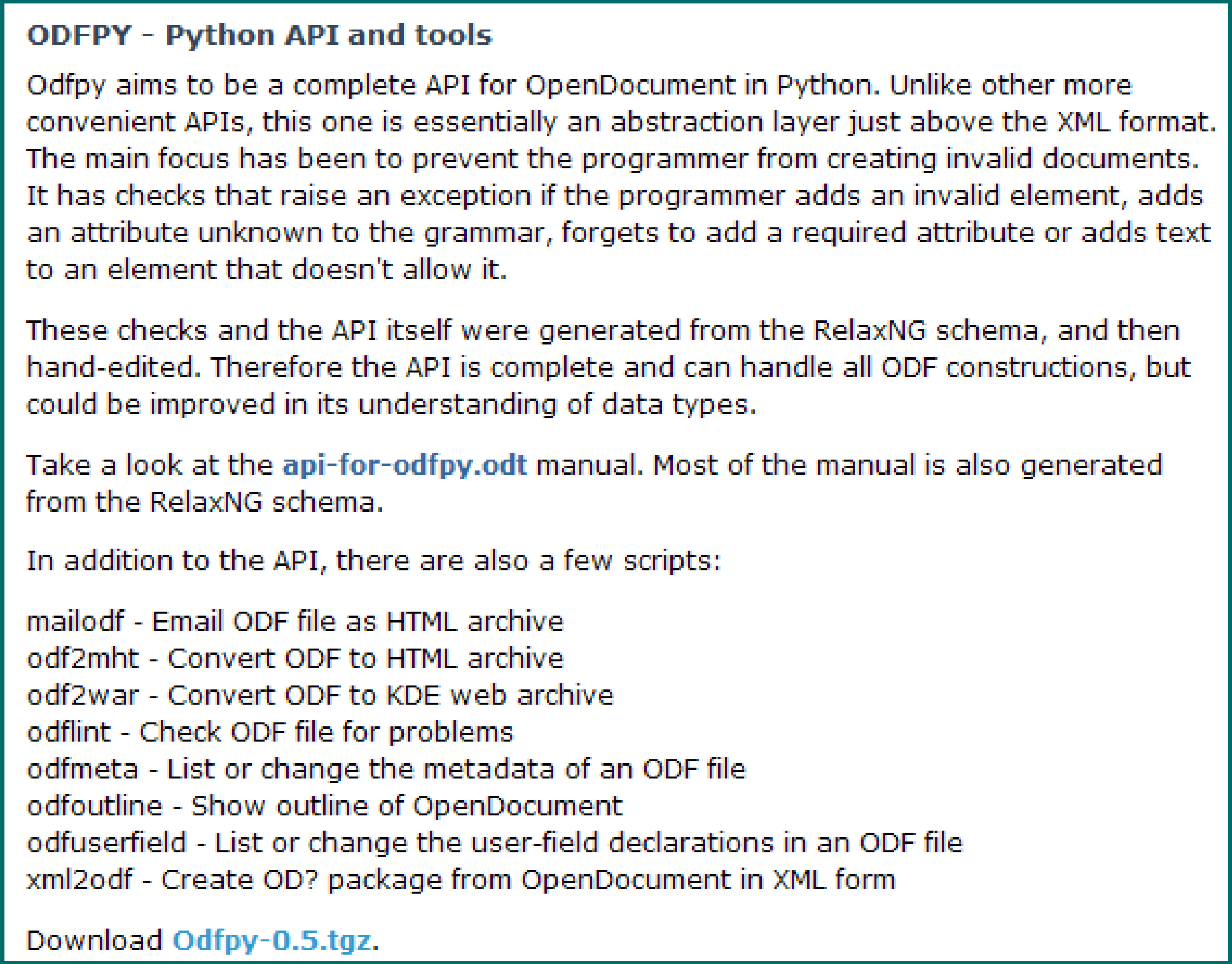
What you end up with

- A family of Parsers and Serializers which can be treated polymorphically (pluggable), using a common ODF DOM representation
- Could become the preferred way to manipulate all office-like documents, not just ODF
- Makes choice of file format irrelevant from the perspective of the application developer
  - Creating an app that supports ODF & Office is the same cost as creating one that supports only Office
  - Reduces switching costs == greater ODF adoption

Things that we can build on

- OpenOffice.org UNO API's
- Apache POI (http://jakarta.apache.org/poi/)
  - Java code for reading/writing MS Office binary formats
- Apache FOP can render to PDF and SVG
- OpenDocument Fellowship has
  - ODT to HTML
  - DocBook to ODT
- J. David Eisenberg has some code in XSLT, Java and Ruby (http://books.evc-cit.info/odf_utils/)
- Probably many others

Odfpy



ODFPY - Python API and tools

Odfpy aims to be a complete API for OpenDocument in Python. Unlike other more convenient APIs, this one is essentially an abstraction layer just above the XML format. The main focus has been to prevent the programmer from creating invalid documents. It has checks that raise an exception if the programmer adds an invalid element, adds an attribute unknown to the grammar, forgets to add a required attribute or adds text to an element that doesn't allow it.

These checks and the API itself were generated from the RelaxNG schema, and then hand-edited. Therefore the API is complete and can handle all ODF constructions, but could be improved in its understanding of data types.

Take a look at the api-for-odfpy.odt manual. Most of the manual is also generated from the RelaxNG schema.

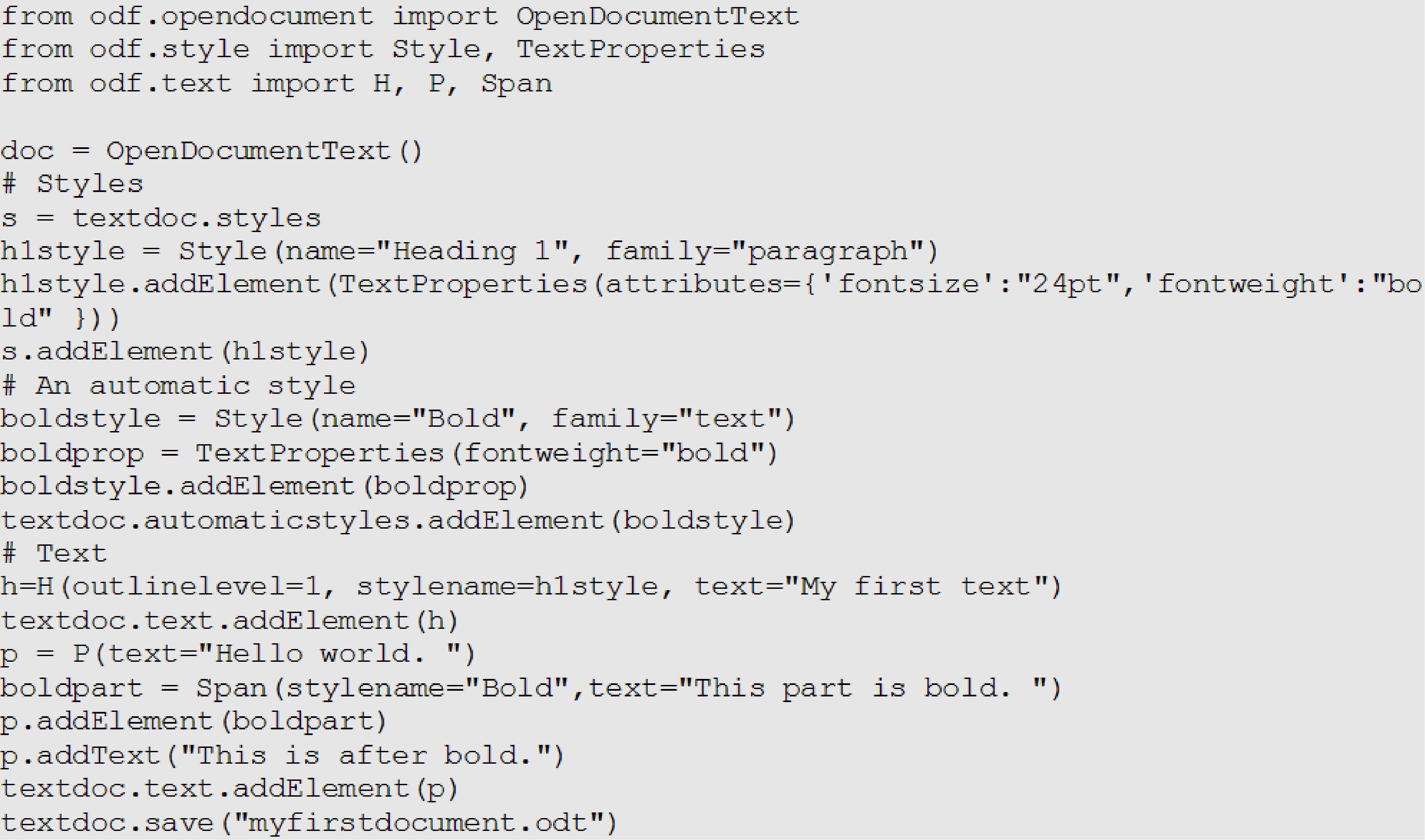
In addition to the API, there are also a few scripts:

mailodf - Email ODF file as HTML archive
odf2mht - Convert ODF to HTML archive
odf2war - Convert ODF to KDE web archive
odflint - Check ODF file for problems
odfmeta - List or change the metadata of an ODF file
odfoutline - Show outline of OpenDocument
odfuserfield - List or change the user-field declarations in an ODF file
xml2odf - Create OD? package from OpenDocument in XML form

Download Odfpy-0.5.tgz.

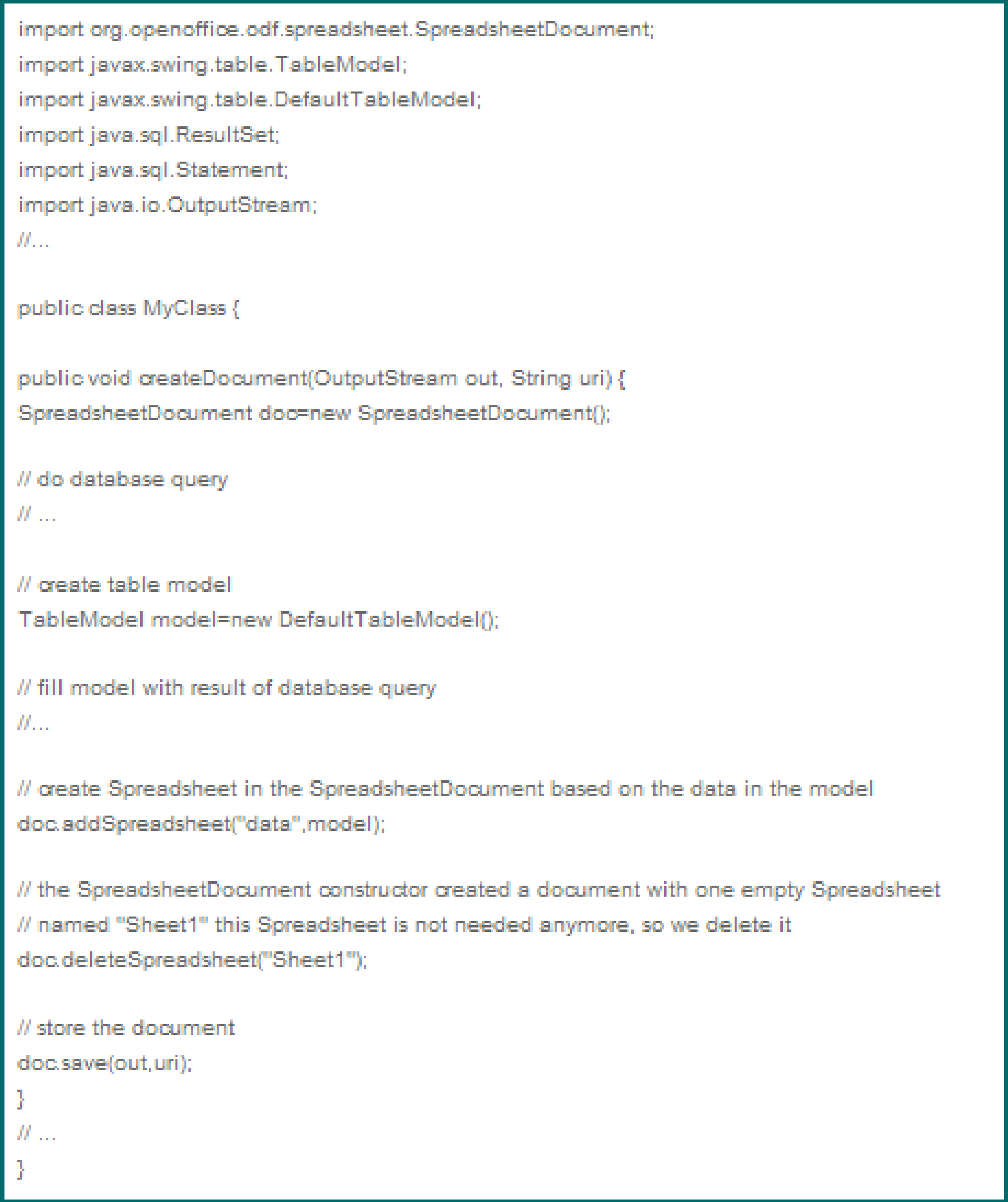
Low Level, close to the XML
Maps validity errors into runtime exceptions.

Odfpy



```
from odf.opendocument import OpenDocumentText
from odf.style import Style, TextProperties
from odf.text import H, P, Span

doc = OpenDocumentText()
# Styles
s = textdoc.styles
h1style = Style(name="Heading 1", family="paragraph")
h1style.addElement(TextProperties(attributes={'fontsize':"24pt",'fontweight':"bo
ld" }))
s.addElement(h1style)
# An automatic style
boldstyle = Style(name="Bold", family="text")
boldprop = TextProperties(fontweight="bold")
boldstyle.addElement(boldprop)
textdoc.automaticstyles.addElement(boldstyle)
# Text
h=H(outlinelevel=1, stylename=h1style, text="My first text")
textdoc.text.addElement(h)
p = P(text="Hello world. ")
boldpart = Span(stylename="Bold",text="This part is bold. ")
p.addElement(boldpart)
p.addText("This is after bold.")
textdoc.text.addElement(p)
textdoc.save("myfirstdocument.odt")
```

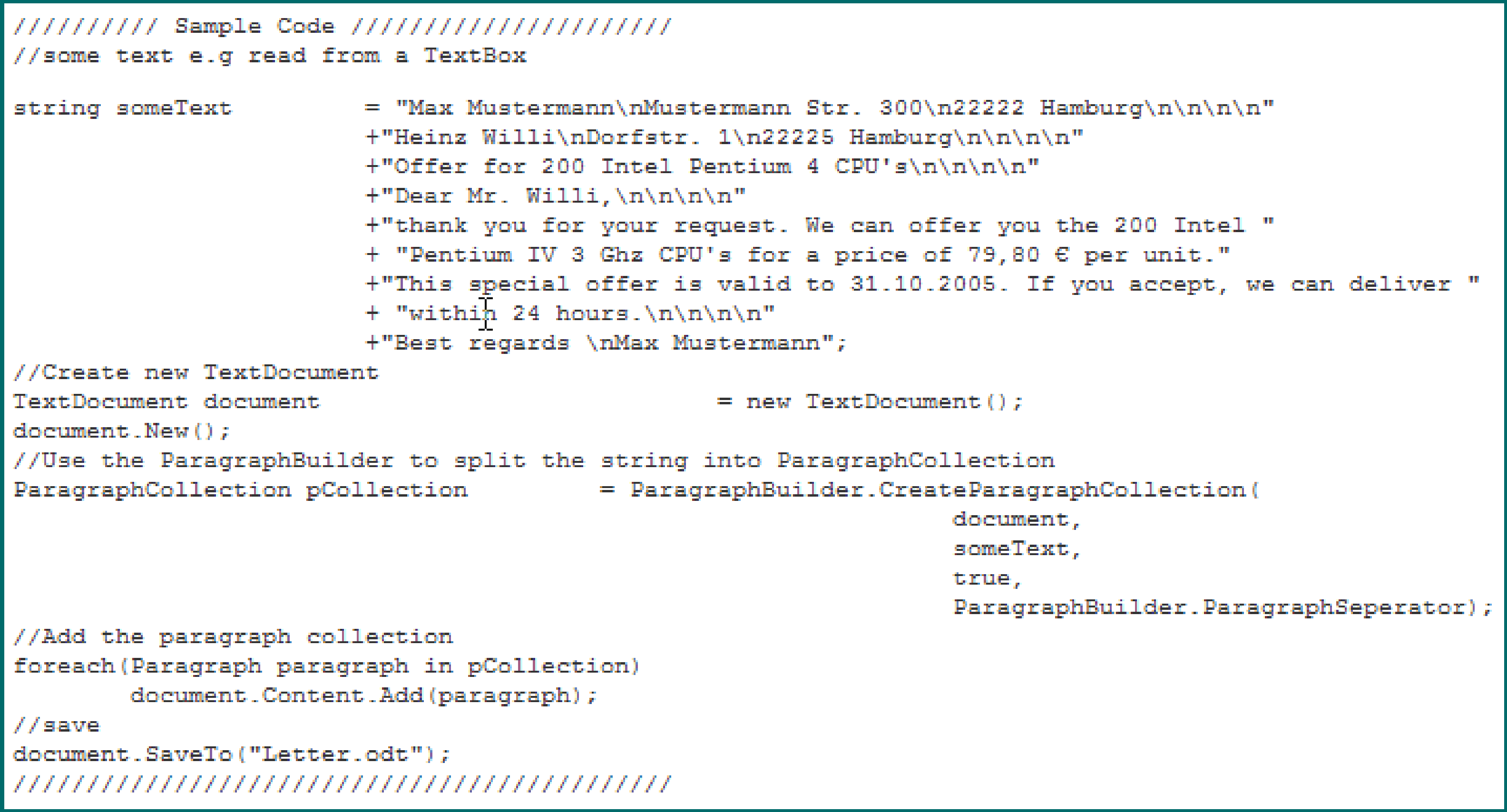
odf4j



```
import org.openoffice.odf.spreadsheet.SpreadsheetDocument;
import javax.swing.table.TableModel;
import javax.swing.table.DefaultTableModel;
import java.sql.ResultSet;
import java.sql.Statement;
import java.io.OutputStream;
//...

public class MyClass {

public void createDocument(OutputStream out, String uri) {
SpreadsheetDocument doc=new SpreadsheetDocument();

// do database query
// ...

// create table model
TableModel model=new DefaultTableModel();

// fill model with result of database query
//...

// create Spreadsheet in the SpreadsheetDocument based on the data in the model
doc.addSpreadsheet("data",model);

// the SpreadsheetDocument constructor created a document with one empty Spreadsheet
// named "Sheet1" this Spreadsheet is not needed anymore, so we delete it
doc.deleteSpreadsheet("Sheet1");

// store the document
doc.save(out,uri);
}
// ...
}
```

Part of OpenOffice.org Toolkit project. Still early.

AODL



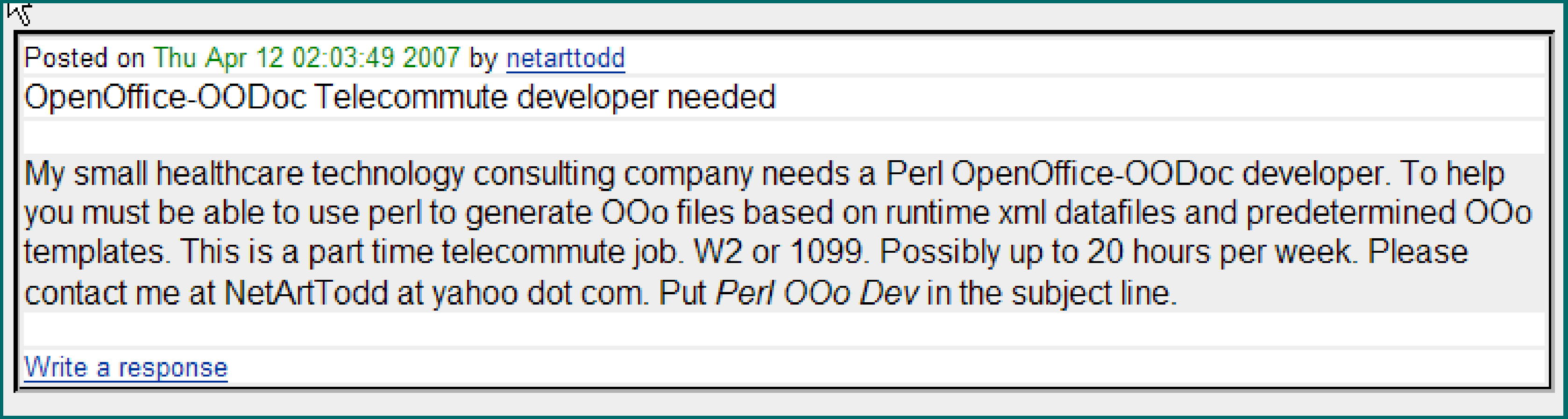
```
////////// Sample Code //////////////////////
//some text e.g read from a TextBox

string someText         = "Max Mustermann\nMustermann Str. 300\n22222 Hamburg\n\n\n\n"
                        +"Heinz Willi\nDorfstr. 1\n22225 Hamburg\n\n\n\n"
                        +"Offer for 200 Intel Pentium 4 CPU's\n\n\n\n"
                        +"Dear Mr. Willi,\n\n\n\n"
                        +"thank you for your request. We can offer you the 200 Intel "
                        + "Pentium IV 3 Ghz CPU's for a price of 79,80 € per unit."
                        +"This special offer is valid to 31.10.2005. If you accept, we can deliver "
                        + "within 24 hours.\n\n\n\n"
                        +"Best regards \nMax Mustermann";
//Create new TextDocument
TextDocument document                           = new TextDocument();
document.New();
//Use the ParagraphBuilder to split the string into ParagraphCollection
ParagraphCollection pCollection         = ParagraphBuilder.CreateParagraphCollection(
                                                                document,
                                                                someText,
                                                                true,
                                                                ParagraphBuilder.ParagraphSeperator);

//Add the paragraph collection
foreach(Paragraph paragraph in pCollection)
        document.Content.Add(paragraph);
//save
document.SaveTo("Letter.odt");
/////////////////////////////////////////////
```

An Open Document Library – C# Library

OpenOffice::OODoc

The Perl Open OpenDocument Connector.

Relatively complete and established.



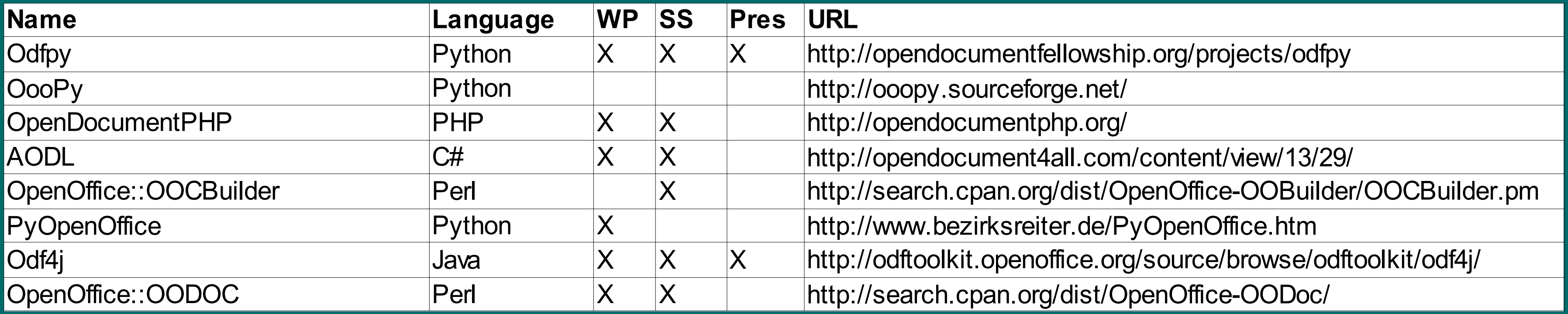
Posted on Thu Apr 12 02:03:49 2007 by netarttodd

OpenOffice-OODoc Telecommute developer needed

My small healthcare technology consulting company needs a Perl OpenOffice-OODoc developer. To help you must be able to use perl to generate OOo files based on runtime xml datafiles and predetermined OOo templates. This is a part time telecommute job. W2 or 1099. Possibly up to 20 hours per week. Please contact me at NetArtTodd at yahoo dot com. Put *Perl OOo Dev* in the subject line.

Write a response

Toolkits I've Looked At



Name	Language	WP	SS	Pres	URL
Odfpy	Python	X	X	X	http://opendocumentfellowship.org/projects/odfpy
OooPy	Python				http://ooopy.sourceforge.net/
OpenDocumentPHP	PHP	X	X		http://opendocumentphp.org/
AODL	C#	X	X		http://opendocument4all.com/content/view/13/29/
OpenOffice::OOCBuilder	Perl		X		http://search.cpan.org/dist/OpenOffice-OOBuilder/OOCBuilder.pm
PyOpenOffice	Python	X			http://www.bezirksreiter.de/PyOpenOffice.htm
Odf4j	Java	X	X	X	http://odftoolkit.openoffice.org/source/browse/odftoolkit/odf4j/
OpenOffice::OODOC	Perl	X	X		http://search.cpan.org/dist/OpenOffice-OODoc/

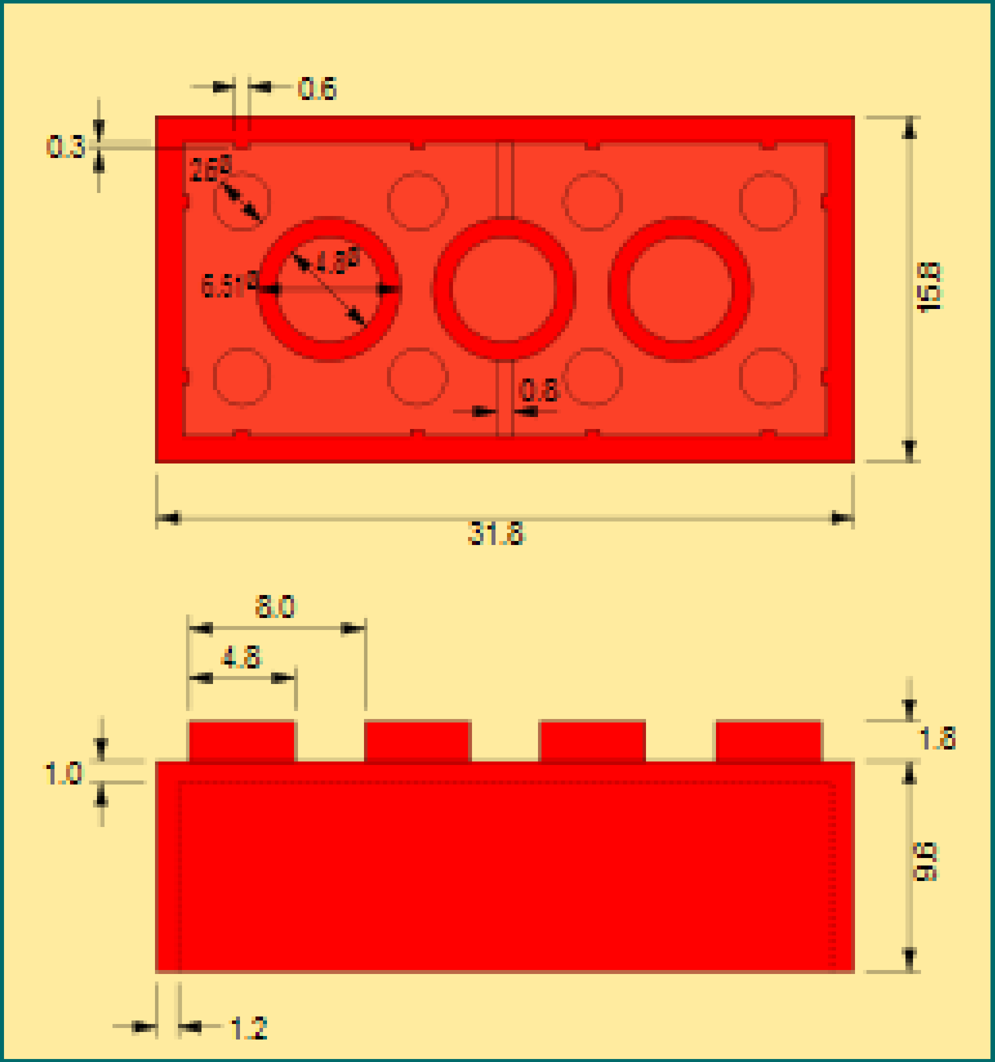
ODF Interoperability

What is Interoperability?

“Interoperability means the ability of information and communication technology (ICT) systems and of the business processes they support to exchange data and to enable the sharing of information and knowledge.”

IDABC's “European Interoperability Framework”
http://ec.europa.eu/idabc/servlets/Doc?id=19529

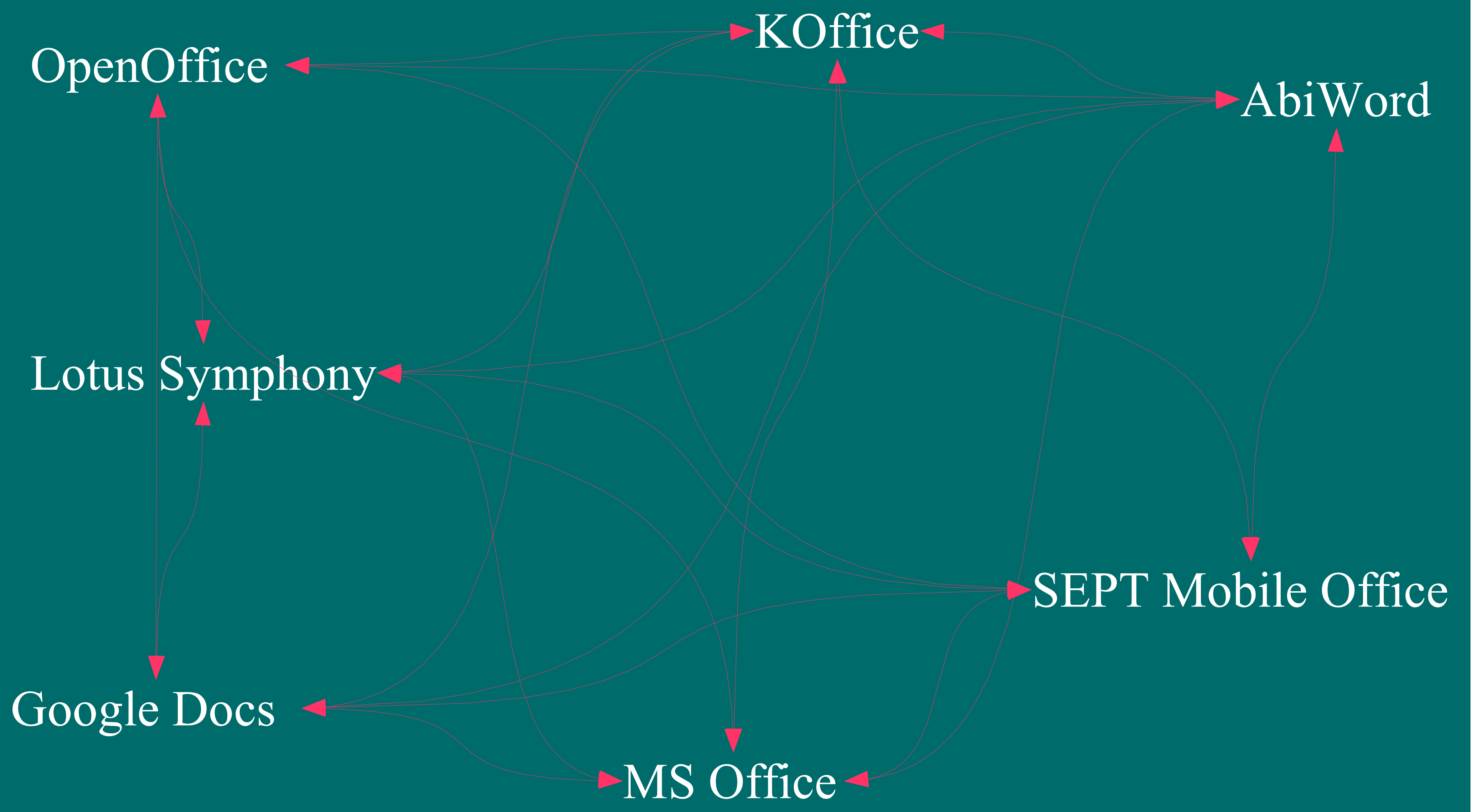
Legos – the intuitive example



Interoperable since 1958.

0.002mm tolerances.

Many ODF Implementations



KOffice

OpenOffice

AbiWord

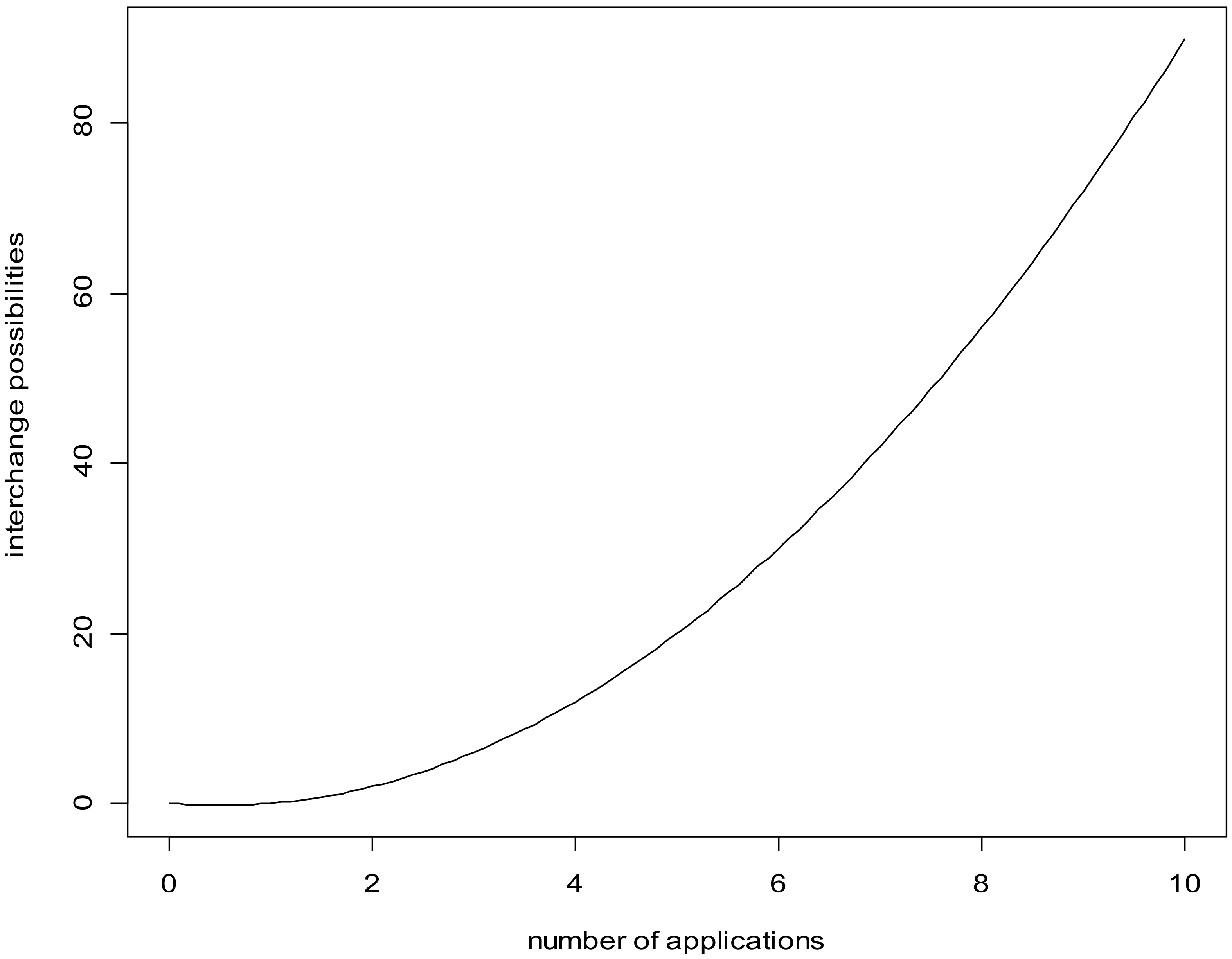
Lotus Symphony

SEPT Mobile Office

Google Docs

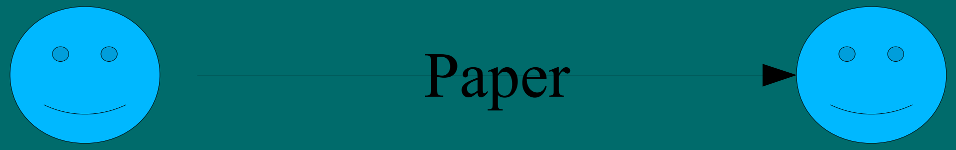
MS Office

With N editors, there are N(N-1) interoperability paths: 2, 6,12, 20, 30, 42, 56, 72,90*


interchange possibilities
0
20
40
60
80
0
2
4
6
8
10
number of applications

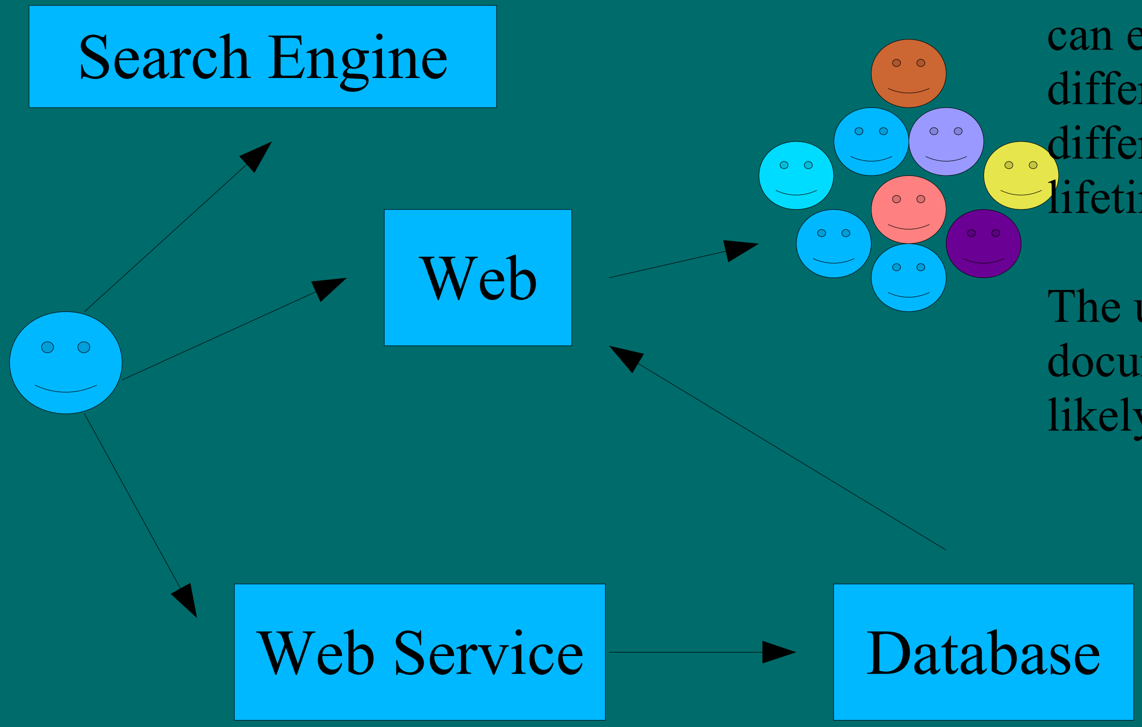
And don't forget the non-editors

Before:



Paper

Now:



Search Engine

Web

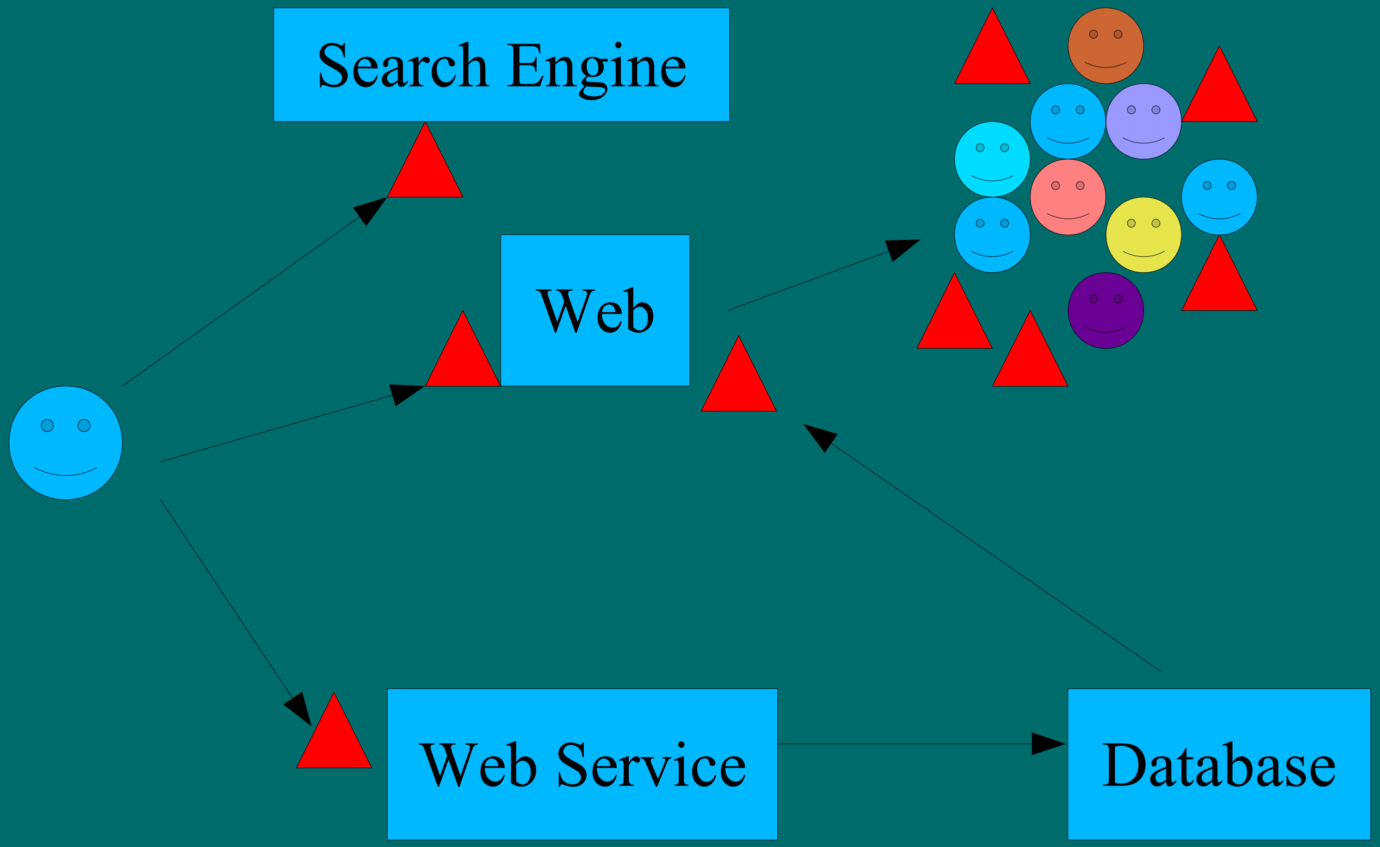
Web Service

Database

A single document can easily be touched by a dozen different applications from different vendors during its lifetime.

The ultimate destination of your document is unknown to you and likely unknowable.

The Interoperability Tax

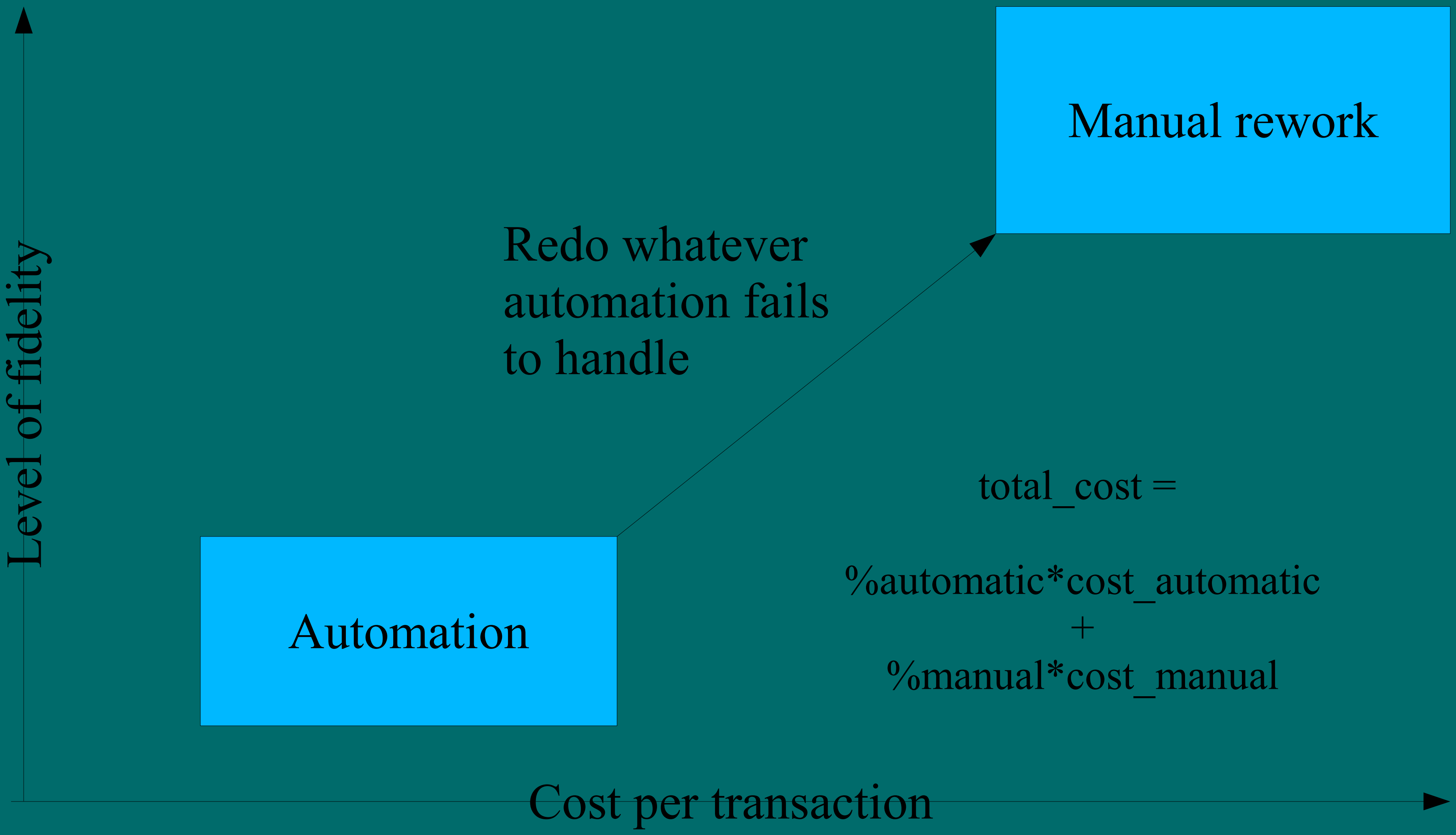


Losses may be:

- Fidelity
- Data
- Performance
- User frustration
- Reputation
- Opportunity

▲ = processing step with loss caused by poor interoperability

*Perfect Interoperability is Easy**

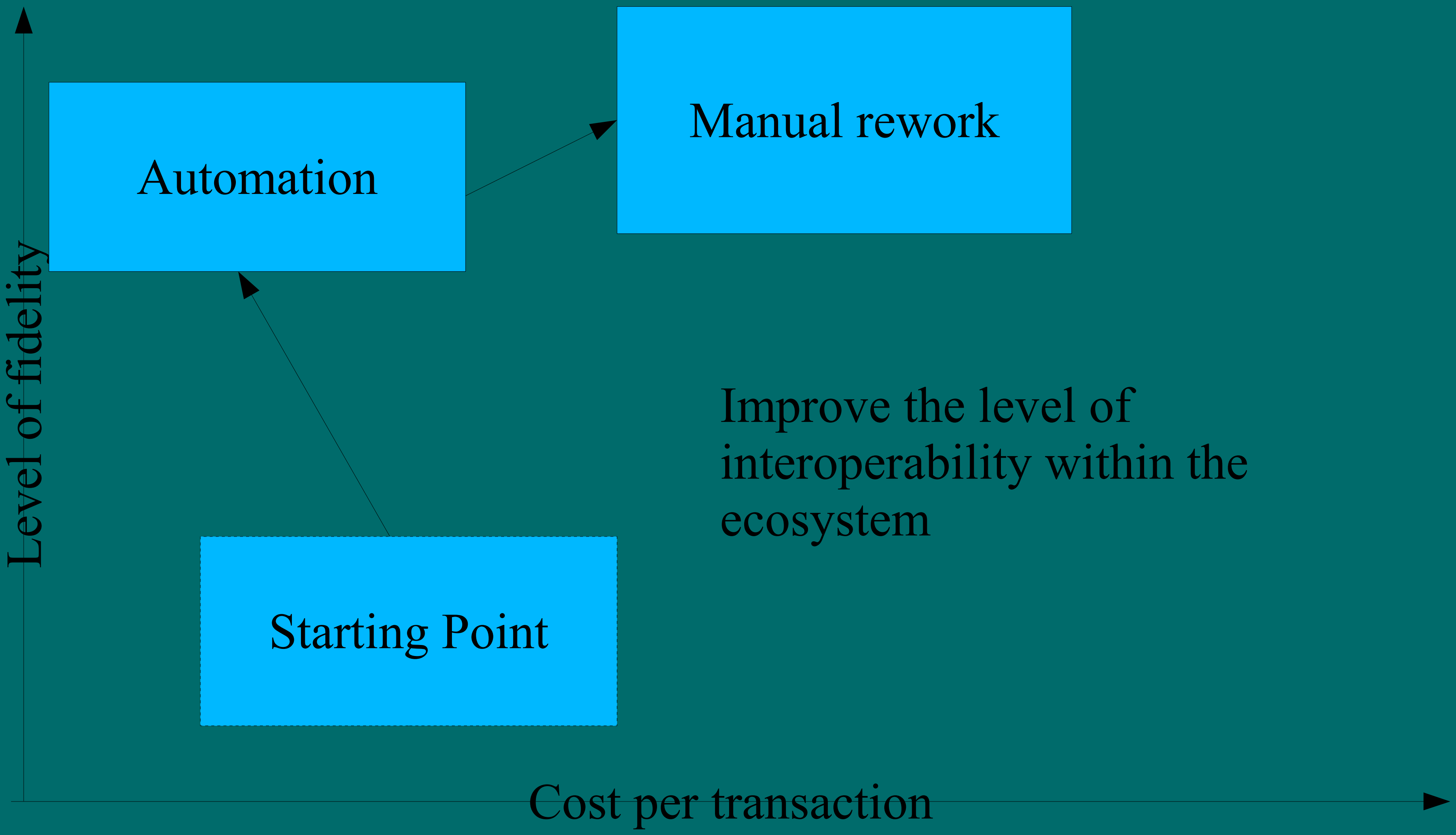


total_cost =

%automatic*cost_automatic
+
%manual*cost_manual

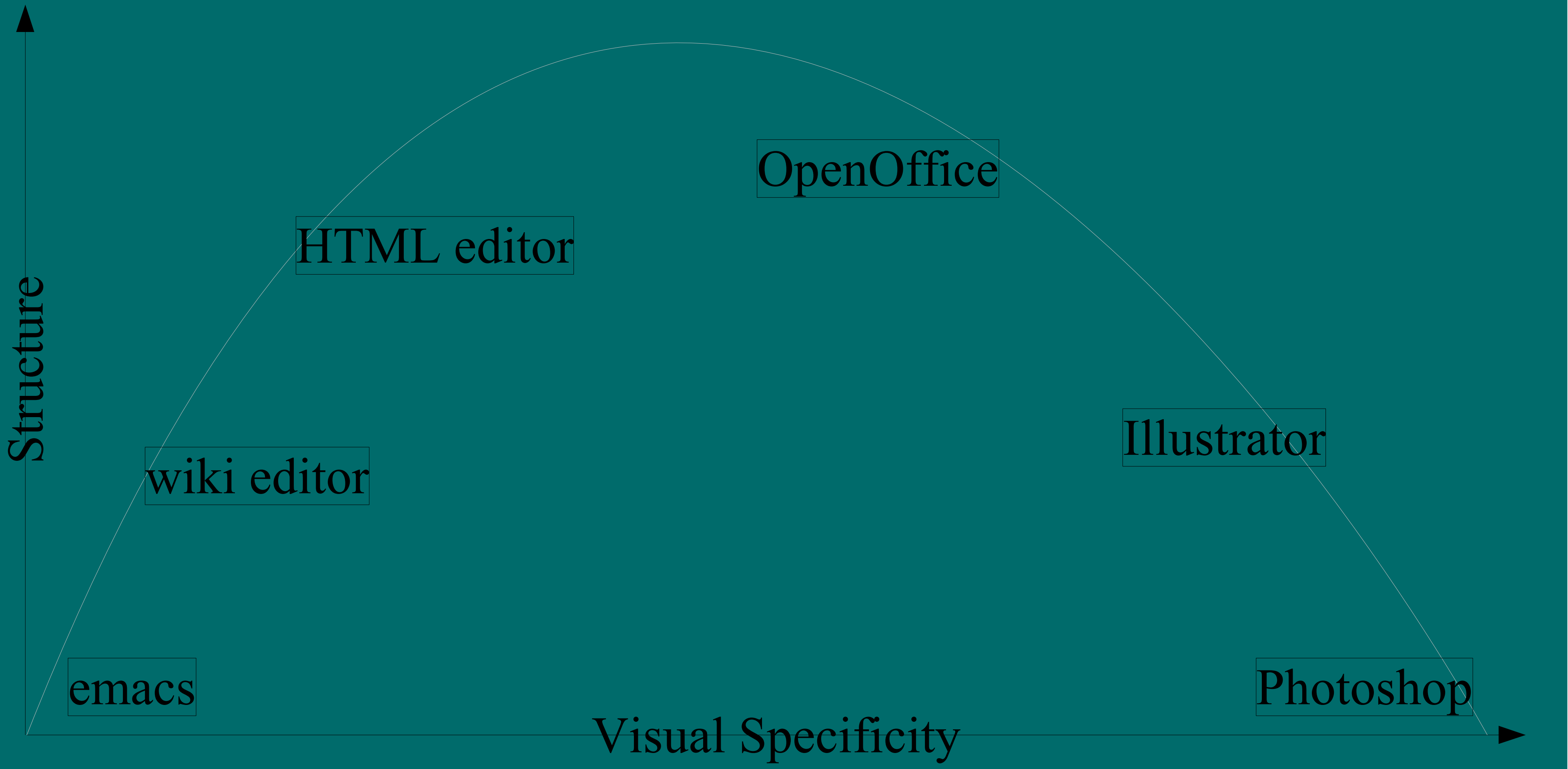
* But expensive

The Goal

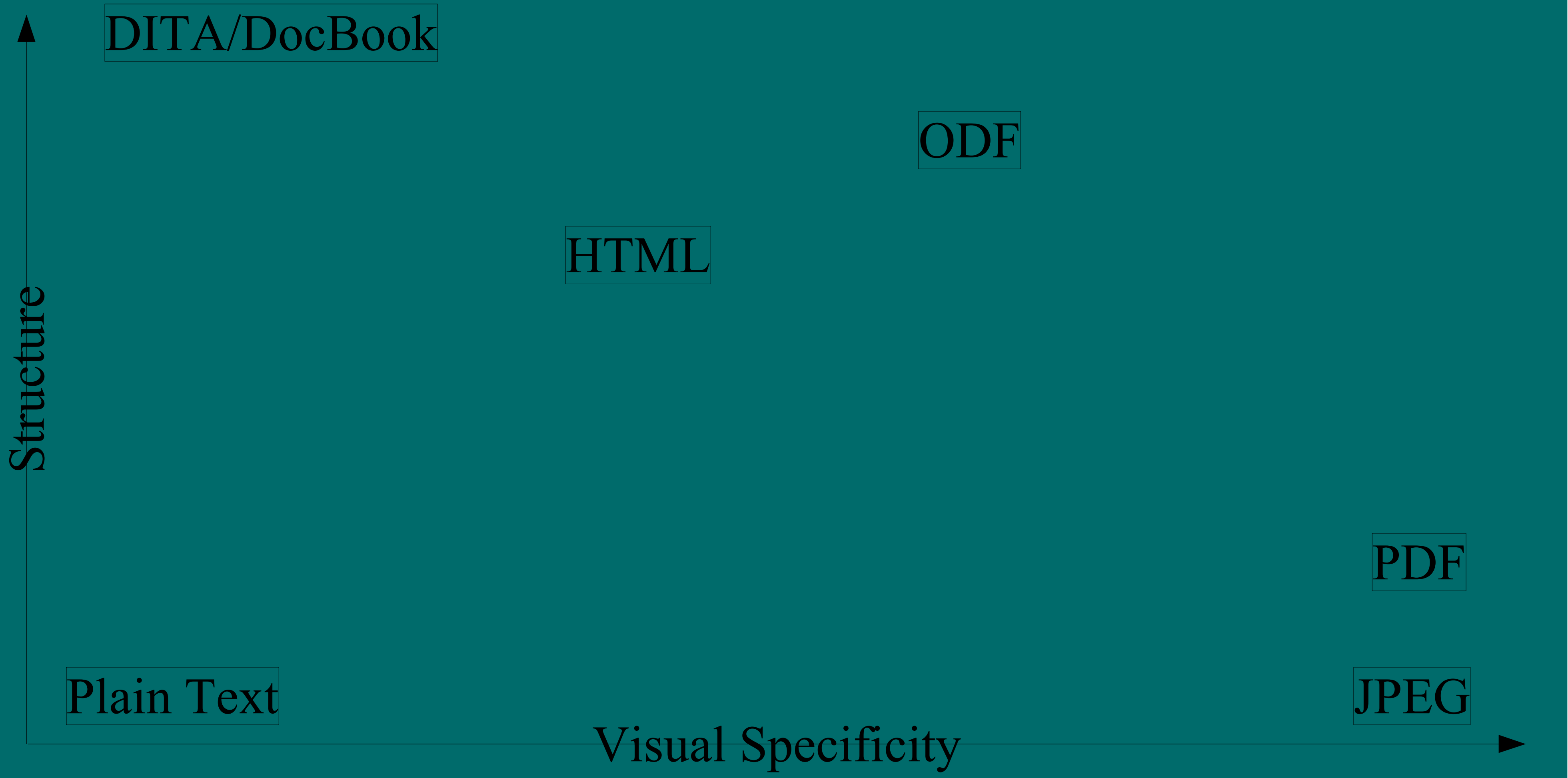


Improve the level of interoperability within the ecosystem

A range of available editors



And a range of formats



DITA/DocBook

ODF

HTML

Structure

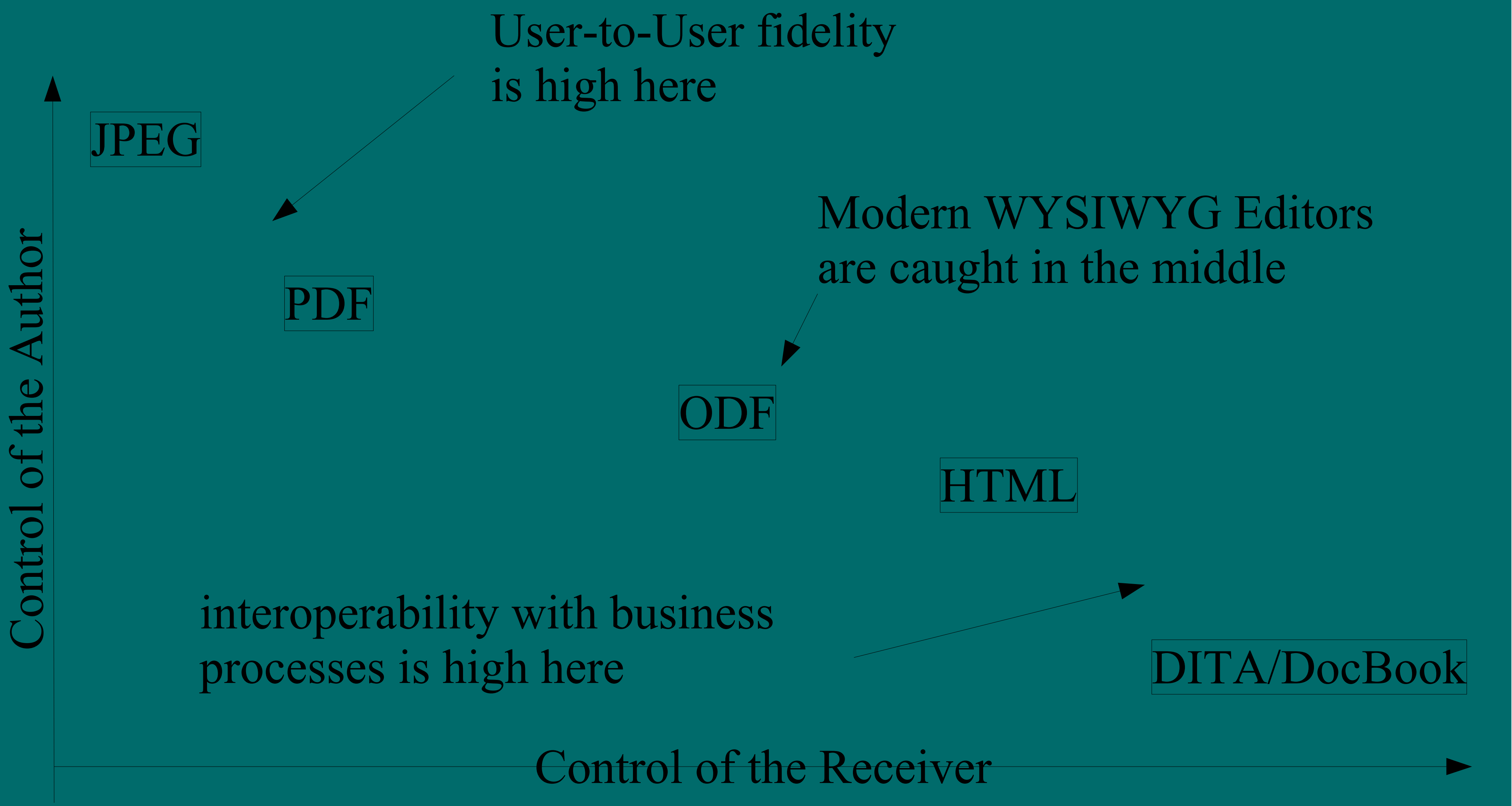
PDF

Plain Text

JPEG

Visual Specificity

And in terms of control...



User-to-User fidelity is high here

Modern WYSIWYG Editors are caught in the middle

interoperability with business processes is high here

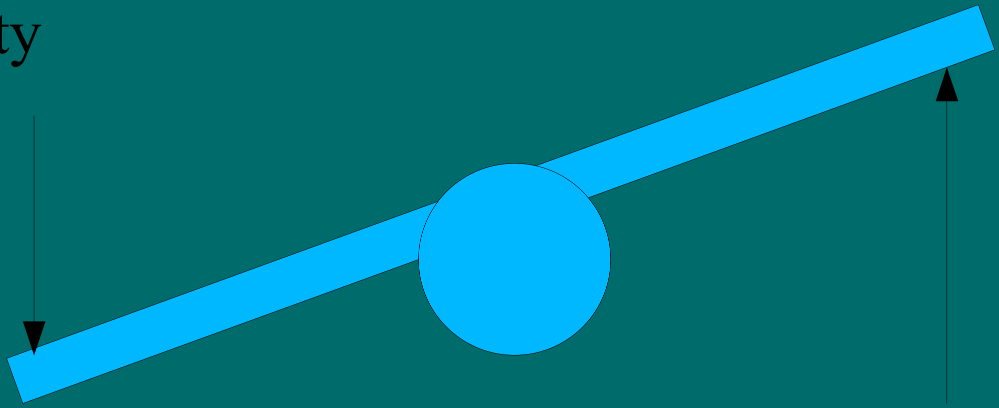
So what do you emphasize?

- Modern word processor has evolved into a multi-paradigm tool that supports different styles of use:
  - Highly structured data oriented use
  - Ad-hoc, visually-oriented layout
- Users have expectations that word processors are suited for both uses. Until the last person who ever used a typewriter is dead, this will continue.

Traditional Trade-offs

1.Visual Richness of authoring environment
2.Power
3.Ability to say anything
4.Pixel Perfection
5.High Fidelity

1.Accessibility
2.Universality
3.Ability of everyone to understand
4.Structure
5.Semantic richness

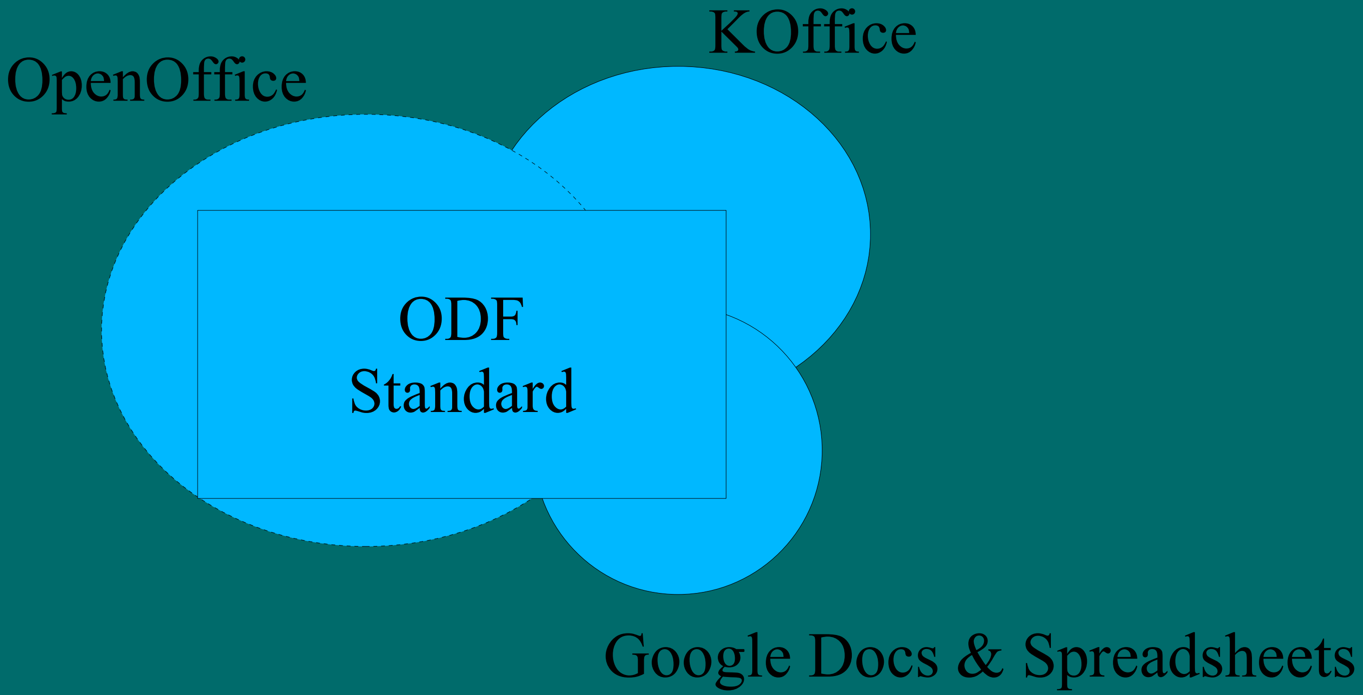


Not a Law of Nature, but a tendency. The glory goes to those who can solve both problems at once.

Things that cause problems

- Application issues
  - Implementation defects
  - Functional subsets
  - Functional supersets (extensions)
- Standard issues
  - Specification errors
  - Undefined behaviors
  - Implementation-defined behaviors

The Conundrum

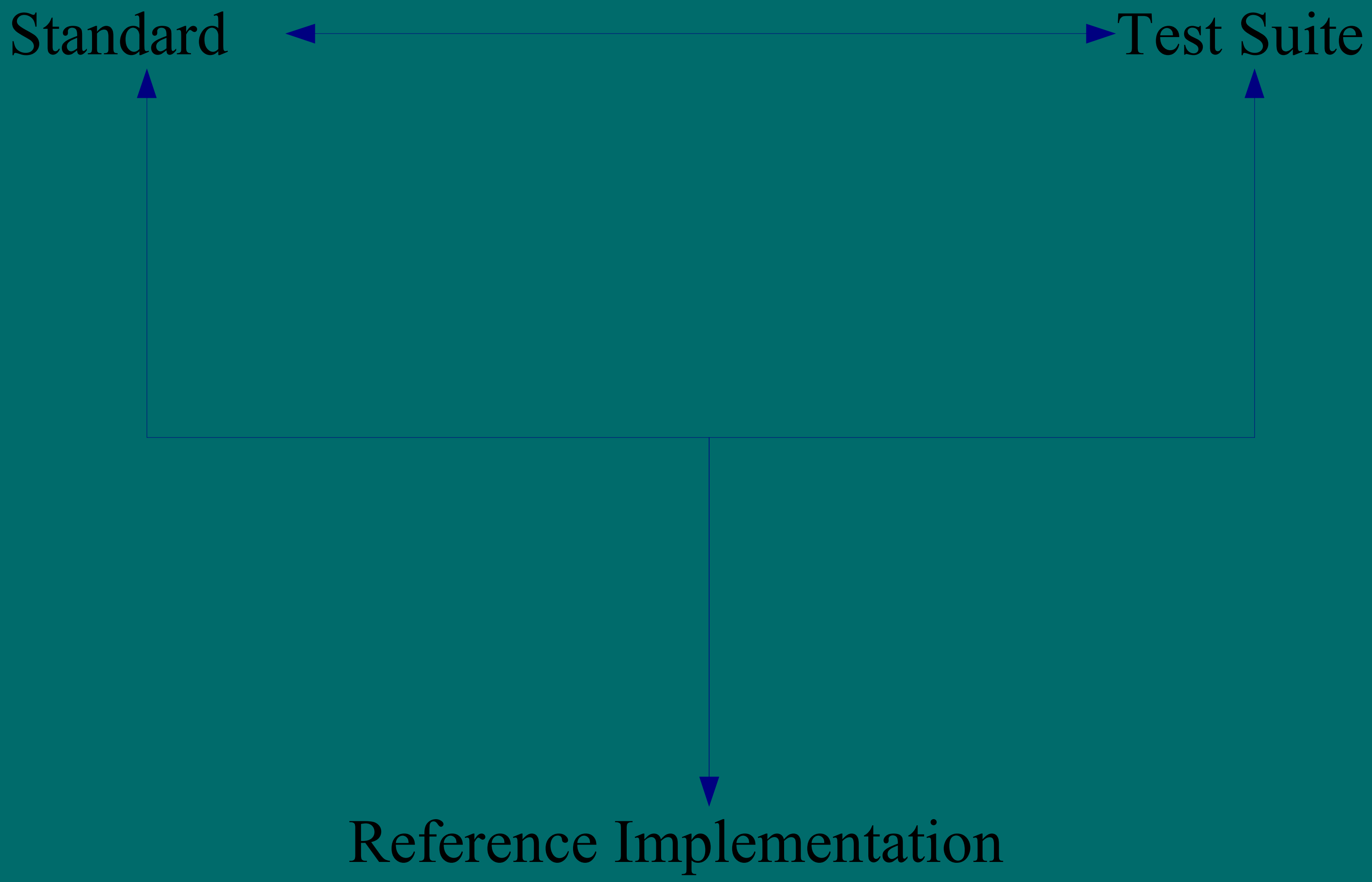


What is the effective overlap?

Solution Patterns

- Standards-development
  - Multi-vendor, multi-stakeholder participation
  - Expert review
  - Implementation concurrent with standards development
- Standards
  - Detailed conformance clauses
  - Deep schemas, allowing deep validation
  - Reference implementations
- Post-standardization activities
  - Translation of standard
  - Development of conformance assessments
  - Multiple implementations

A powerful pattern



Standard

Test Suite

Reference Implementation

A powerful pattern

- The standard contains the definition of a conformant document
  - (but the standard might have errors or ambiguities)
- The test suite exercises and validates each feature of the standard
  - (but the test suite might have errors or omissions)
- The reference implementation is written to the standard, and tested with the test suite
  - (but the implementation might have errors or missing functionality)

Checks and Balances

- A test case fails. What is the cause?
  - An error in the application?
  - Is it an error in the test suite?
  - An error in the standard?
- Identify the cause of the failure
- Fix
- Continue until you have a complete test suite and a reference implementation that passes all of the test cases.

A Reference Implementation

- Should implement 100% of the standard, including all optional requirements.
- It should be the first one, or one of the first applications to implement any new feature in the standard.
- For any implementation-defined behaviors, it should document how it behaves.
- Although it may extend the standard, it should have a mode of operation where it is strictly conformant.

A Test Suite: A rough estimate

- ~ 700 page ODF specification
- ~ 5 testable statements per page
- ~ 4 test cases per statement to test limits, positive and negative test cases, etc.

- So, on the order of 10,000 test cases, or 2 PY of effort.

Things that foster interoperability

- In applications:
  - use of interoperable data formats
  - a strictly conforming mode of operation
  - guidance to the user on how to use the product in an interoperable way
  - inclusion of document templates and defaults that encourage interoperability
  - allowing validation of documents
- In data formats
  - clean separation of content, attributes, behavior and metadata
  - reuse of existing, established standards
  - thorough review
  - standardization

Things that foster interoperability

- In organizations:
  - adoption of a single standard document format
  - adoption of applications with proven conformance to that document standard
  - training of users on how to create interoperable documents

- In users:
  - capture information at the highest level possible
  - adding metadata
  - providing annotations for accessibility
  - using named styles

Progress in Interoperability

- Test Suites
- Validators
- Translators

ODF Test Suite

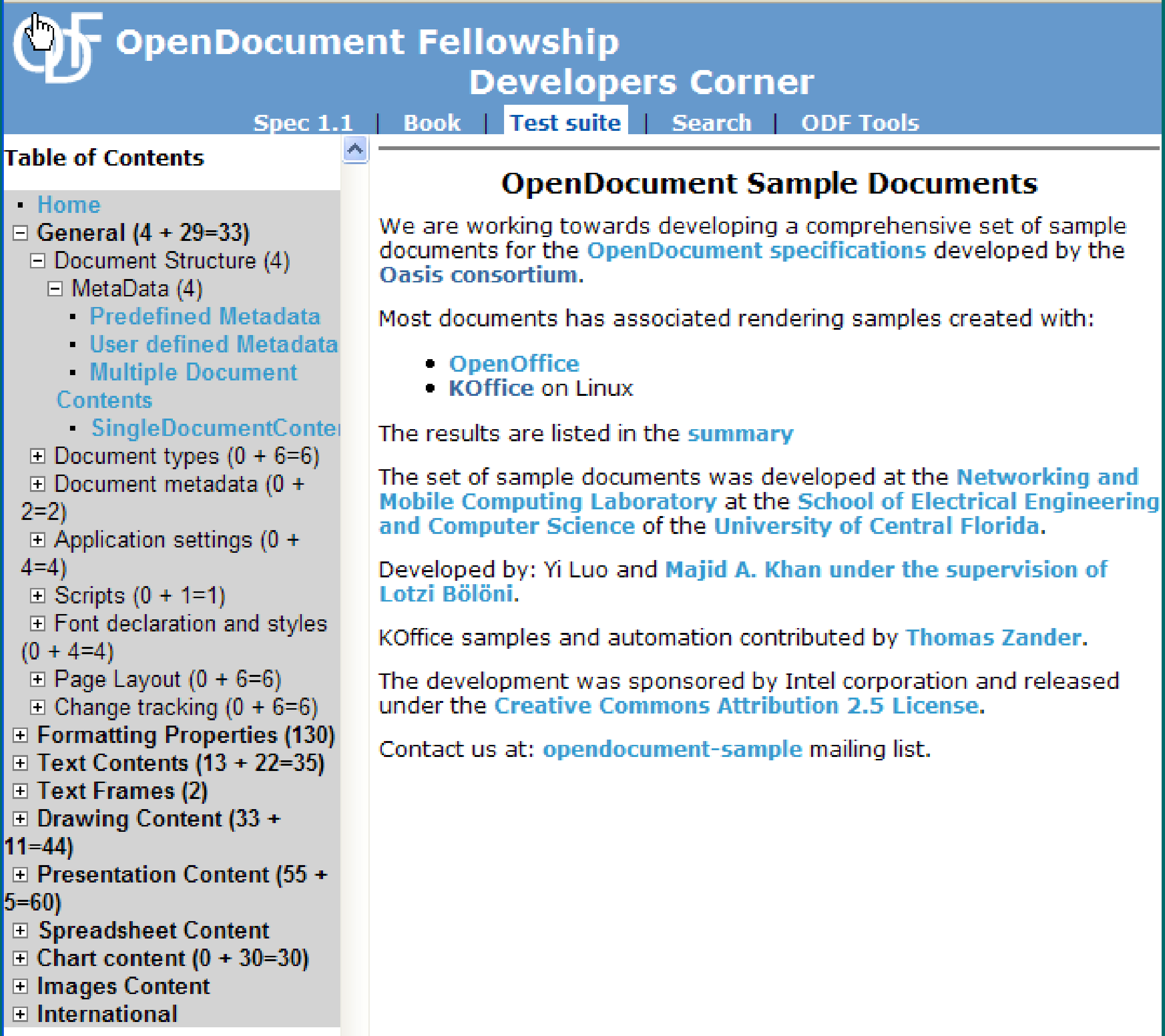


Table of Contents

- Home
- ⊟ General (4 + 29=33)
  - ⊟ Document Structure (4)
    - ⊟ MetaData (4)
      - Predefined Metadata
      - User defined Metadata
      - Multiple Document Contents
      - SingleDocumentConte
  - ⊞ Document types (0 + 6=6)
  - ⊞ Document metadata (0 + 2=2)
  - ⊞ Application settings (0 + 4=4)
  - ⊞ Scripts (0 + 1=1)
  - ⊞ Font declaration and styles (0 + 4=4)
  - ⊞ Page Layout (0 + 6=6)
  - ⊞ Change tracking (0 + 6=6)
- ⊞ Formatting Properties (130)
- ⊞ Text Contents (13 + 22=35)
- ⊞ Text Frames (2)
- ⊞ Drawing Content (33 + 11=44)
- ⊞ Presentation Content (55 + 5=60)
- ⊞ Spreadsheet Content
- ⊞ Chart content (0 + 30=30)
- ⊞ Images Content
- ⊞ International

OpenDocument Sample Documents

We are working towards developing a comprehensive set of sample documents for the OpenDocument specifications developed by the Oasis consortium.

Most documents has associated rendering samples created with:

- OpenOffice
- KOffice on Linux

The results are listed in the summary

The set of sample documents was developed at the Networking and Mobile Computing Laboratory at the School of Electrical Engineering and Computer Science of the University of Central Florida.

Developed by: Yi Luo and Majid A. Khan under the supervision of Lotzi Bölöni.

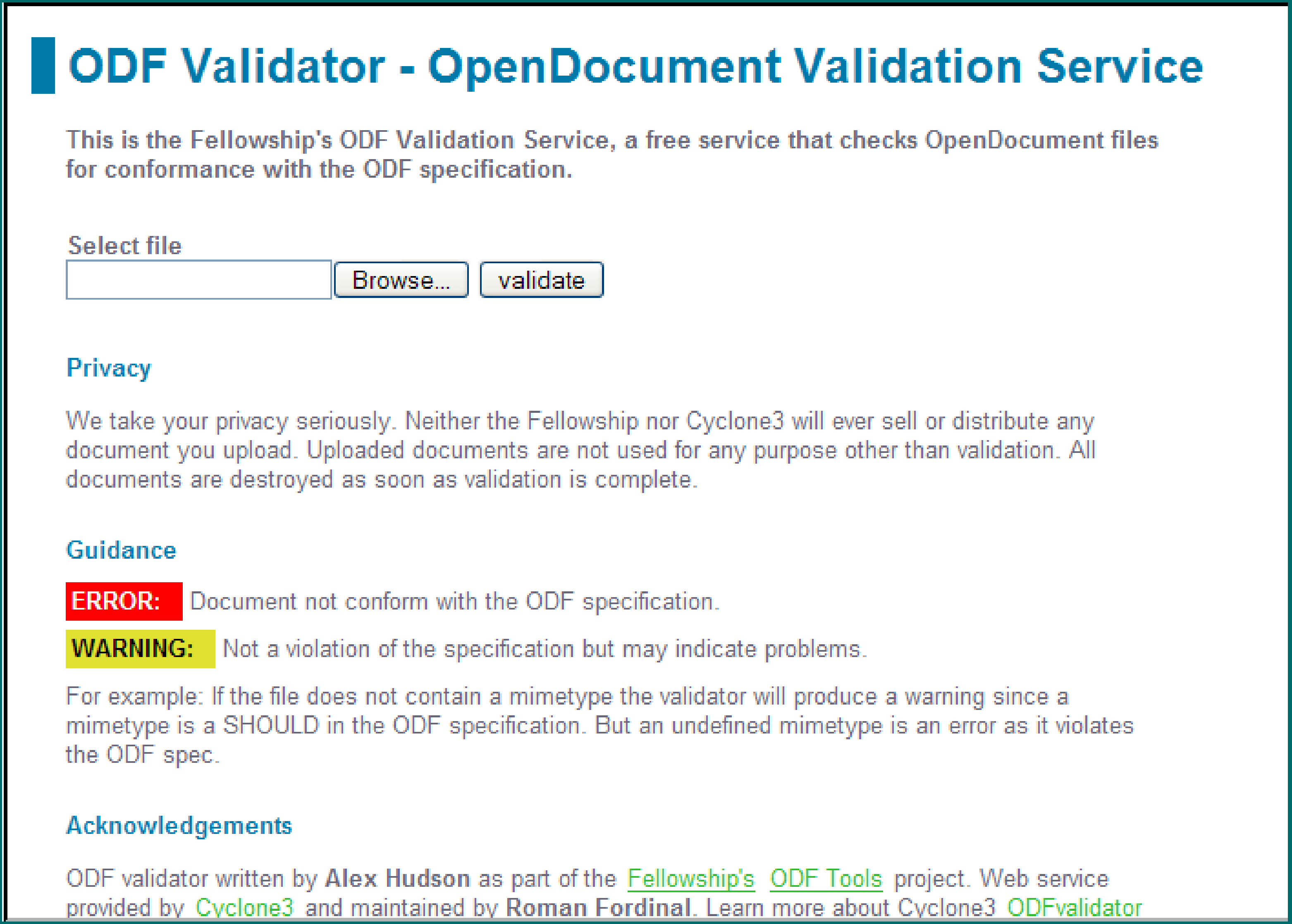
KOffice samples and automation contributed by Thomas Zander.

The development was sponsored by Intel corporation and released under the Creative Commons Attribution 2.5 License.

Contact us at: opendocument-sample mailing list.

http://develop.opendocumentfellowship.org/testsuite/

ODF Validator



ODF Validator - OpenDocument Validation Service

This is the Fellowship's ODF Validation Service, a free service that checks OpenDocument files for conformance with the ODF specification.

Privacy

We take your privacy seriously. Neither the Fellowship nor Cyclone3 will ever sell or distribute any document you upload. Uploaded documents are not used for any purpose other than validation. All documents are destroyed as soon as validation is complete.

Guidance

ERROR: Document not conform with the ODF specification.

WARNING: Not a violation of the specification but may indicate problems.

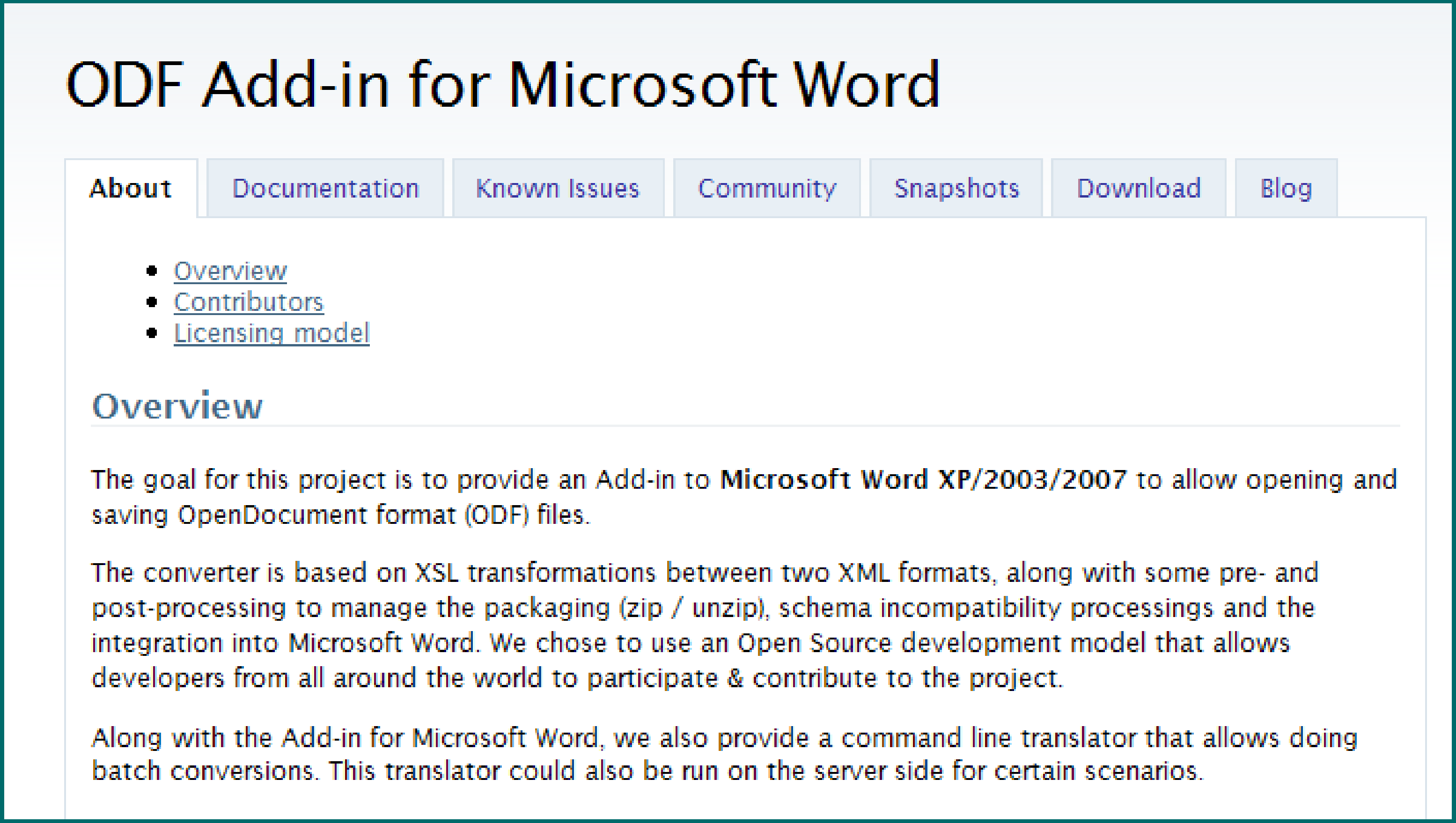
For example: If the file does not contain a mimetype the validator will produce a warning since a mimetype is a SHOULD in the ODF specification. But an undefined mimetype is an error as it violates the ODF spec.

Acknowledgements

ODF validator written by **Alex Hudson** as part of the Fellowship's ODF Tools project. Web service provided by Cyclone3 and maintained by **Roman Fordinal**. Learn more about Cyclone3 ODFvalidator

http://opendocumentfellowship.org/validator

ODF Add-in for Word



ODF Add-in for Microsoft Word

- Overview
- Contributors
- Licensing model

Overview

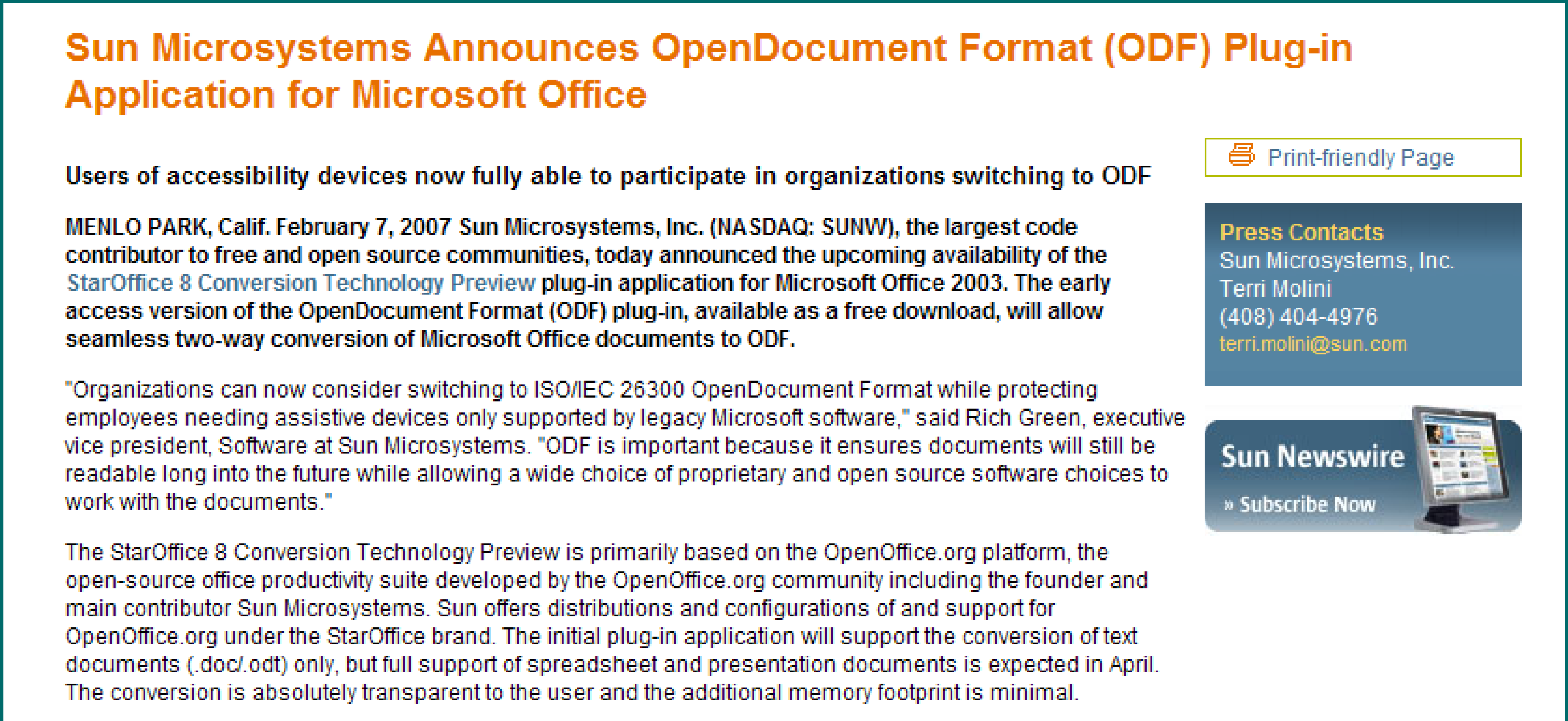
The goal for this project is to provide an Add-in to **Microsoft Word XP/2003/2007** to allow opening and saving OpenDocument format (ODF) files.

The converter is based on XSL transformations between two XML formats, along with some pre- and post-processing to manage the packaging (zip / unzip), schema incompatibility processings and the integration into Microsoft Word. We chose to use an Open Source development model that allows developers from all around the world to participate & contribute to the project.

Along with the Add-in for Microsoft Word, we also provide a command line translator that allows doing batch conversions. This translator could also be run on the server side for certain scenarios.

http://odf-converter.sourceforge.net/

ODF Plug-in for MS Office



Sun Microsystems Announces OpenDocument Format (ODF) Plug-in Application for Microsoft Office

Users of accessibility devices now fully able to participate in organizations switching to ODF

MENLO PARK, Calif. February 7, 2007 Sun Microsystems, Inc. (NASDAQ: SUNW), the largest code contributor to free and open source communities, today announced the upcoming availability of the StarOffice 8 Conversion Technology Preview plug-in application for Microsoft Office 2003. The early access version of the OpenDocument Format (ODF) plug-in, available as a free download, will allow seamless two-way conversion of Microsoft Office documents to ODF.

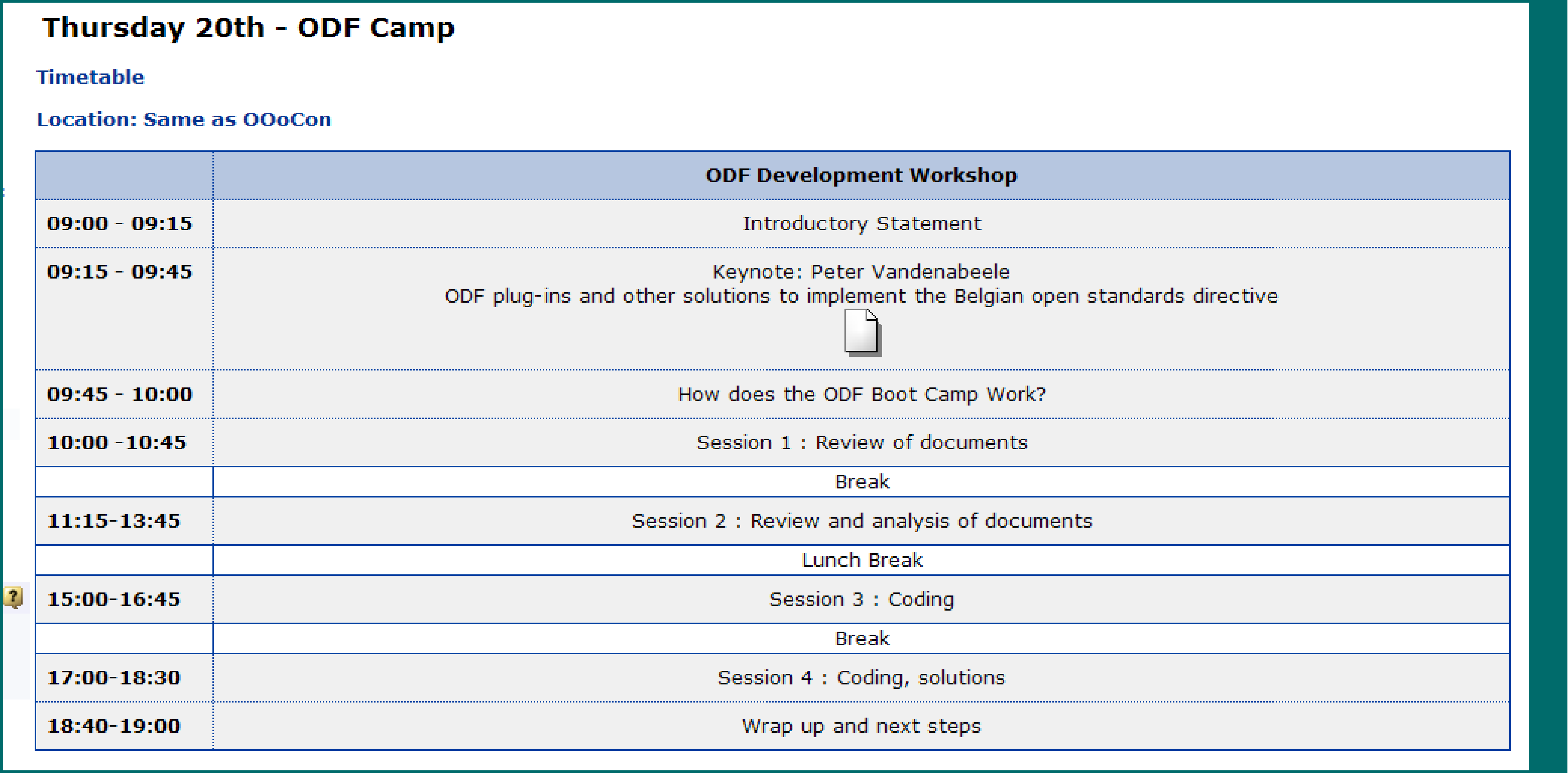
"Organizations can now consider switching to ISO/IEC 26300 OpenDocument Format while protecting employees needing assistive devices only supported by legacy Microsoft software," said Rich Green, executive vice president, Software at Sun Microsystems. "ODF is important because it ensures documents will still be readable long into the future while allowing a wide choice of proprietary and open source software choices to work with the documents."

The StarOffice 8 Conversion Technology Preview is primarily based on the OpenOffice.org platform, the open-source office productivity suite developed by the OpenOffice.org community including the founder and main contributor Sun Microsystems. Sun offers distributions and configurations of and support for OpenOffice.org under the StarOffice brand. The initial plug-in application will support the conversion of text documents (.doc/.odt) only, but full support of spreadsheet and presentation documents is expected in April. The conversion is absolutely transparent to the user and the additional memory footprint is minimal.

Press Contacts
Sun Microsystems, Inc.
Terri Molini
(408) 404-4976
terri.molini@sun.com

http://www.sun.com/software/star/openoffice/

ODF Interoperability Camp



Thursday 20th - ODF Camp

Timetable

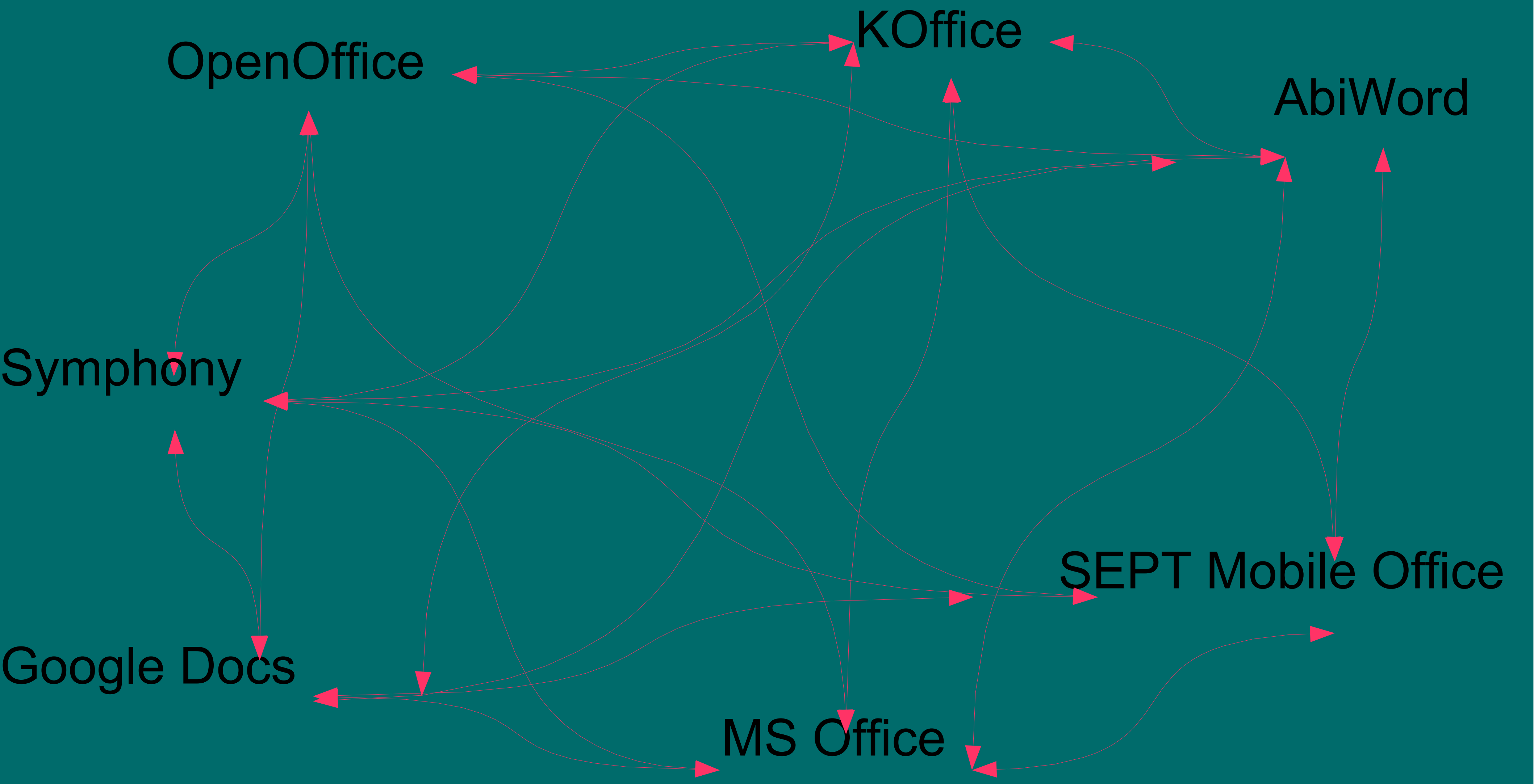
Location: Same as OOoCon

	ODF Development Workshop
09:00 - 09:15	Introductory Statement
09:15 - 09:45	Keynote: Peter Vandenabeele ODF plug-ins and other solutions to implement the Belgian open standards directive
09:45 - 10:00	How does the ODF Boot Camp Work?
10:00 -10:45	Session 1 : Review of documents
	Break
11:15-13:45	Session 2 : Review and analysis of documents
	Lunch Break
15:00-16:45	Session 3 : Coding
	Break
17:00-18:30	Session 4 : Coding, solutions
18:40-19:00	Wrap up and next steps

Proposed ODF IIC TC

- Implementation
- Interoperability
- Conformance

We must reduce this problem...



KOffice

OpenOffice

AbiWord

Symphony

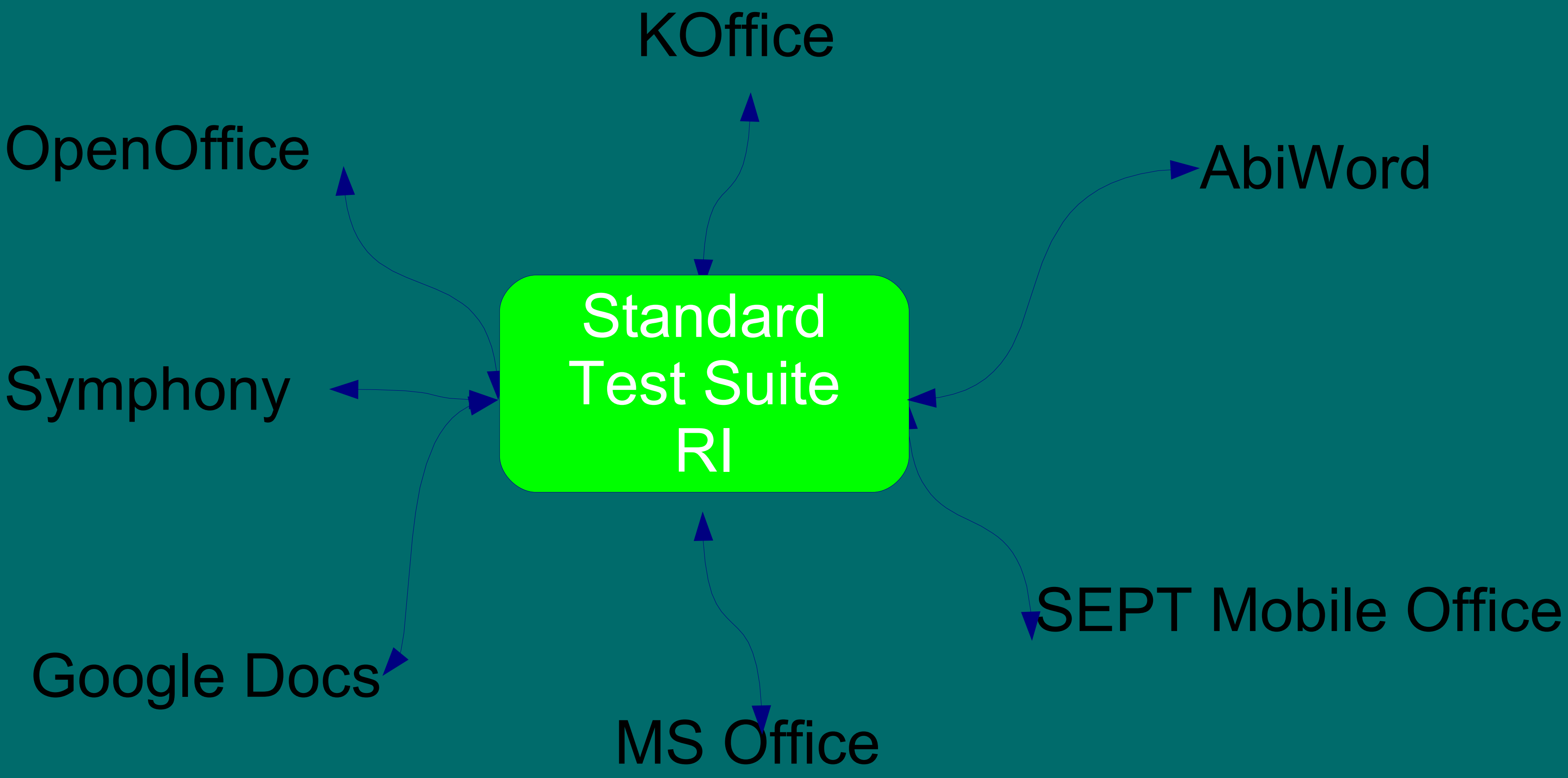
SEPT Mobile Office

Google Docs

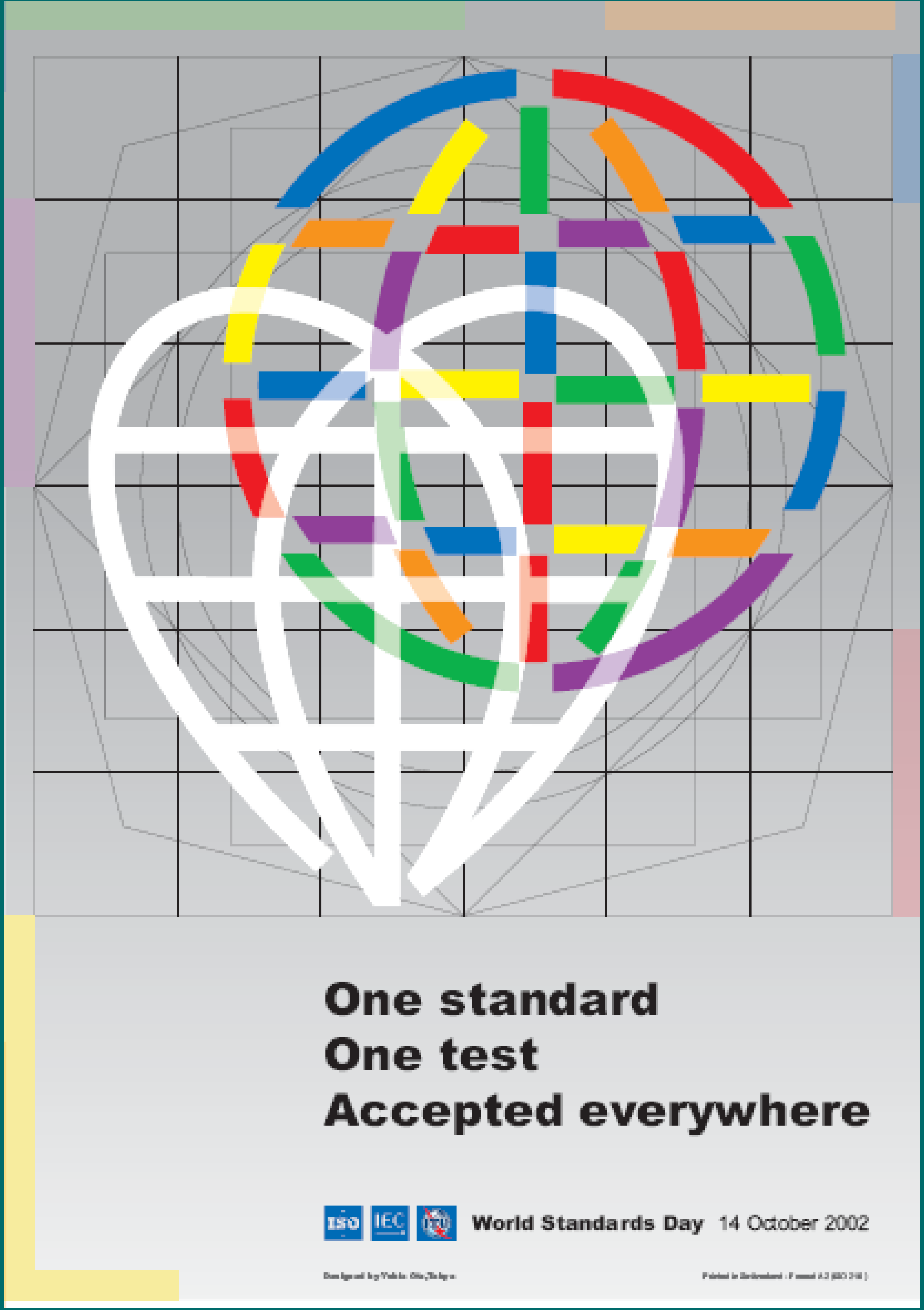
MS Office

With N editors, there are N(N-1) interoperability paths*

To this problem (which we know how to solve)



With N editors, there are N interoperability tests



One standard
One test
Accepted everywhere

ISO IEC ITU **World Standards Day** 14 October 2002

Designed by Yukio Ota/Tokyo

Printed in Switzerland · Format A3 (ISO 216)