Specification of "Layout-true Representation of OOXML Documents in Open Source Office Applications"

Version 6, December 12th 2011

Open Source Business Alliance (OSBA) Working Group
"Office Interoperability"

Table of Contents

Introduction

This specification document forms the basis on which software companies can develop tenders for the technical implementation of the specification requirements. The specification itself is based on the outcomes of a workshop held in Zürich between October 10th and 11th 2011, in which public institutions and LibreOffice/OpenOffice developers together decided the requirements that shall be included.

Objective

The following requirements aim to improve the accuracy of OOXML document layout in LibreOffice/OpenOffice. A seamless integration of the OOXML specification into LibreOffice/OpenOffice is not being sought. Instead, a pragmatic approach is desired that, by making specific corrections to the OOXML filter and other software components in LibreOffice/OpenOffice as necessary, provides as true as possible a visualization of OOXML documents as they appear in Microsoft Office 2007/2010, together with the corresponding user benefits.

Organization within the OSBA Working Group Framework

This project will be managed by the "Office Interoperability" Working Group of the Open Source Business Alliance (OSBA). The Working Group is open to all members of the OSBA. Interested parties should contact the Working Group spokesperson.

The Working Group members together ensure the funding required for the implementation work. They appoint a spokesperson, who acts as the central point of contact for all project communication and coordination matters. The Working Group spokesperson releases the specification to the media for publication, receives tenders from the software companies, follows the process of appointing a contractor, prepares the master agreement, coordinates financing agreements and the signing of contractual agreements, monitors implementation progress and follows the acceptance of the software development. However, the spokesperson does not have the authority to sign any contract in the name of the Working Group. Orders are placed by Working Group members and other financial backers directly with the contractor.

Accordingly, the implementation work is divided into suitable areas and is appointed by the Working Group members and other financial backers in the form of direct orders with the chosen contractor. A single contractor will perform the implementation work due to the clear scope of work involved and also to manage coordination costs. A standard master agreement or harmonized individual contract concluded between all financing institutions and the contractor shall guarantee that the services are properly coordinated and that a functioning end product will be delivered.

Schedule (subject to change)

12th December 2011	Publication of specification and test documents (access on request)
31st January 2012	Deadline for tenders from software companies
29th February 2012	End of selection process, contractor is appointed
31st March 2012	All contracts are signed, implementation can begin
31st May 2012	All requirements have been implemented, tested and accepted
30th June 2012	Source code is published under an Open Source license and integrated into the LibreOffice project. The project is concluded.

General Conditions

The general conditions govern the overriding stipulations for implementing the requirements.

Terminology

- Microsoft Word, Microsoft Excel and Microsoft PowerPoint refer to both 2007 and 2010 versions of the program.
- OOXML is the general term used in this document for the new .docx, .xlsx and .pptx XML-based document formats introduced in Microsoft Office 2007.
- LibreOffice/OpenOffice is the term used to represent all Open Source Office applications based on OpenOffice.org.
- A "true representation" is an accurate positioning of all pixels at a resolution of 70 dpi.
- "Valid" is a validation of the XML for ODF 1.0/11 as per the "strict" requirements and for ODF 1.2 as per the "conformant" requirements.

Test Documents

The use case test documents act as reference files to aid the contractor when testing the implementation work and shall apply as an acceptance criterion for each respective use case. These are normally .docx files containing certain layout elements as well as corresponding PDF files (exported from Microsoft Office) that portray the desired appearance. If a test document is opened using the improved OOXML representation in an Open Source Office application and exported as a PDF, it should appear as largely identical to the reference PDF.

Compliance with the requirements will depend on a pixel comparison between the test documents and the PDF export from the improved OOXML filter. The pixel comparison shall be carried out by the contractor using Officeshots[1] or a comparable verification method.

Furthermore, LibreOffice/OpenOffice must be able to save the test document as a valid ODF and reload it without the loss of any data. Should ODF XML upgrades be needed to achieve this, these must be suggested to the OASIS ODF TC. Validity shall then include these proposed ODF upgrades. Validity must be checked using the validator provided by the Apache ODF Toolkit[2] or an equivalent validator.

The test documents are saved in the corresponding folders under a common directory on Filespots[3] Interested contractors should contact the Working Group spokesperson to obtain access to the directory.

Code Basis

The software must be developed based on the code basis of the latest OOXML filter in LibreOffice. This filter is currently the best option for implementing the OOXML standard in an Open Source environment.

License

The contractor is obliged to publish the new OpenOffice/LibreOffice software developed for this project under Apache License Version 2.0. This license permits the widest possible distribution of the development services.

1 http://officeshots.org
2 http://incubator.apache.org/odftoolkit/conformance/ODFValidator.html http://odf-validator.rhcloud.com
3 https://www.filespots.com

Technical Requirements

Detailed specifications of each use case are outlined below. These use cases are a collection of requirements that are similar in nature. Each contains a description of the content as well as several test documents.

Use Case 1: Border and Picture Formatting in .docx

Aims

Borders of tables and paragraphs formatted in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.

1. The size of the border matches the original document
2. The style of the border matches the original document
3. The position of the border matches the original document

Pictures embedded in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.

1. LibreOffice/OpenOffice supports the import, embedding and export of picture formats that are also supported by Microsoft Office.
2. If an OOXML document contains an unknown picture format, a placeholder matching the original picture dimensions is shown instead to maintain the original document layout.

Test Documents

See files in the relevant folder on Filespots.

Use Case 2: Table Formatting in .docx

Aims

1. *Tables* formatted in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.
2. *Nested tables* created in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.
3. Vertical and horizontal *spaces* in and around tables defined in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.

Notes

Microsoft introduced new table design templates with the launch of Office 2007. This made it possible to format tables from a number of designs at the click of a mouse. A special feature here is the differing formatting options for even/odd columns and lines. These design instructions for tables are not hardcoded when saved in OOXML format, which is why the LibreOffice/OpenOffice filter does not know how the table elements (borders, backgrounds and fonts) should be formatted. LibreOffice/OpenOffice must be taught how to convert design classes for tables that are embedded in a .docx file into accurate

LibreOffice/OpenOffice formatting instructions. A pragmatic approach that sees the formatting options hardcoded in LibreOffice/OpenOffice is conceivable.

Nested tables are emulated in LibreOffice/OpenOffice with a border. However, the border cannot be paginated onto the next page. Tables can do this easily, which is why representation and handling problems occur.

Test Documents

See files in the relevant folder on Filespots.

Use Case 3: List Formatting in .docx

Aims

Lists (numbered or otherwise) formatted in Microsoft Word and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.

1. The dimensions (e.g. indents) match the original document in size and appearance.
2. The bullet point symbols match the original document.
3. The numbering convention matches the original document.
4. The nesting matches the original document.
5. The document structure is not changed when importing from the subsequent OOXML format export.

Notes

Problems in displaying bullet points are often related to unavailable fonts. A pragmatic approach that sees the bullet point symbols hardcoded in LibreOffice/OpenOffice is conceivable.

The document structure must not be changed so as to prevent unwanted problems with any automated content that it supports (e.g. tables of contents or similar) as well as VBA macros.

Test Documents

See files in the relevant folder on Filespots.

Use Case 4: Comment Formatting in .docx and .xlsx

Aim

Comments made in Microsoft Word and Microsoft Excel and saved in OOXML format have to be imported as a true representation into LibreOffice/OpenOffice, and subsequently exported back to OOXML.

Notes

The problem with the correct positioning of comments in Calc could be solved by merely introducing a patch to LibreOffice. A porting of the patch would be required were this solution to be chosen.

Test Documents

See files in the relevant folder on Filespots.

Use Case 5: Embedding Fonts in OOXML and ODF

Aims

1. To provide an option whereby non-standard fonts can be embedded in both OOXML and ODF files when saving from LibreOffice/OpenOffice.
2. Embedded fonts in OOXML and ODF files are loaded and displayed correctly when importing the document into LibreOffice/OpenOffice.

Notes

When saving OOXML and ODF files, LibreOffice/OpenOffice must embed font intelligently; i.e. the developer must specify what fonts can be expected on a system as standard. Microsoft Office must be able to read and load fonts exported in OOXML files.

Furthermore, it must be noted that embedded fonts have an effect on the file size. Font files can comprise up to several hundred kilobytes, massively increasing the memory space required to save just one file. This effect must be moderated by an embedding option while saving.

In the medium-term, embedding of fonts will become part of the ODF standard; however, this feature has until now not been included within the scope of the standard. An example of this form of implementation would be a reference to web fonts.

Embedded fonts also carry legal implications - font manufacturers determine in the font files whether a font may be fully or partly embedded in documents, and whether they may be passed on in this form or not (copyright). This is currently the case with many, but not all, fonts. It may also be the case that the font manufacturer only permits a subset to be embedded, i.e. partial embedding. In this scenario, only the characters that are actually used in the text are saved in the document. A substitute font has to be used if a character not contained in the subset is added to the text in an external system, which in turn causes further formatting problems.

Test Documents

See files in the relevant folder on Filespots.

Tendering Procedure

Provider Details

The tender must contain a detailed introduction to the provider, its history, senior management, staff, their abilities and also reference projects and customers. Details on the number of staff, their background and experience (CVs) as well as previous reference projects (customers, scope of project, activities, etc.) are also expected. In addition, the tender must outline the knowledge and expertise concerning LibreOffice/OpenOffice software development, the OOXML and ODF specification, OOXML filter, etc. required for this project, including how and why the provider can demonstrate these skills.

Method

The tender must describe the approach that will be adopted to implement the technical requirements. A solution concept setting out the activities, a schedule and development methodology is expected. Aspects such as testing, documentation, security and stability must all be accounted for. Tenders must also specify how the resulting software source code at the end of the development stage will be incorporated into the main tree of the LibreOffce Open Source project and remain so in further official releases.

Language

Tenders and related correspondence must be written in German or English.

Deadline

Tenders must be submitted by to the Working Group spokesperson by e-mail and in PDF format by January 31st 2012. Any questions or comments may be raised with the designated spokesperson of the "Office Interoperability" Working Group:

Dr. Matthias Stürmer, Ernst & Young
matthias.stuermer@ch.ey.com
+41 58 289 61 97

Selection Procedure

Financing members of the Working Group together with other financial backers shall independently choose the contractor. Should the Working Group feel that no suitable tender has been submitted, the call to tender may be canceled at any time.