Bibliographic Project's Developer Page

Last Modified
2005-December-4

A printer friendly PDF version of this page is available developer1.pdf (36Kb)

Contents

Project Overview

The role of the Bibliographic Project (OOoBib) is to support the OpenOffice.org Writer (wordprocessing) application by enhancing the bibliographic facility. See our Vision statement for details. Our current objection to to design and build OOoBib version 0.1, which will contain the most basic functions for an usable bibligraphic facility.

For an overview of the Bibliographic project's major components and a context diagram see components.html. There is information about the current OpenOffice Bibliographic implementation.

A start has been made to the Specification for this work (see the Projects Specifications folder on the Documents and Files page). Also see a attempt at an analysis of the proposed Bibliographic enhancement components and their relationships.

The best place to start for finding out about development in OpenOffice is the OpenOffice.org For Developers page. An important resource is the Developer's guide which is part of the SDK (software development kit) or available online on at
http://api.openoffice.org/DevelopersGuide/DevelopersGuide.html

The OOo API is based on UNO (Universal Network Objects) is the interface-based component model of OpenOffice.org. UNO offers interpretability between different programming languages, different object models, different machine architectures and different processes; either in a local network or even via the Internet. UNO components can be implemented in and accessed from any programming

language for which a UNO language binding exists. We currently provide several language bindings for UNO which allows to use the API from Java, C++, OpenOffice.org Basic, Python and Common Language Infrastructure (CLI).

1st Stage, Bibliographic Facility Redevelopment

top of page

Summary

As a first step we are plan to implement the most simple changes to the OOo core code (the API basic code, and UNO mappings, but not yet the user interface code) necessary to implement basic support for -

As a first step we will implement the most simple changes to the OOo core code (the API basic code, and UNO mappings, but not yet the user interface code) necessary to implement basic support for:

1. Support saving and reading enhanced citation support in OpenDocument
2. Ability to insert and display citations in OpenOffice Writer using the new format. (Note this task does not include the GUI interface to insert the citation in the new format, only the UNO interface to provide the basic function.
3. Storage of document bibliographic data in the OOo document save package and the code changes necessary to read and save that bibliographic data.

When these basic functions are built into OOo and are made assessable via the UNO, we can then use rapid prototyping development methods to design and build prototype GUI interfaces and bibliographic formatting engines. We will be able to use any of the programming languages which have OpenOffice bindings: C++, Java, Python and, of course, OpenOffice Basic. We believe that we will find more developers who can work in these languages than by insisting on C++ code from the start. Also it is much easier to build prototypes using Java, Python and OpenOffice Basic than in C++.

NB. When we have designed, built and tested the prototypes and they have been accepted by the OOo community we intend to rebuild them in C++ and to have them made part of the core OpenOffice application.

Skills required - good C++ programming and some XML skills with knowledge of, or willingness to learn, the OpenOffice UNO (see the Openoffice Developer's Guide)

---

1. Modify the Writer document-read and document-save modules to support the new OpenDocument enhanced citation format.

Implement the citation and bibliography changes to the OOo Writer save file (in Open Document format) accepted by the OpenDocument Technical Committee. The changes to the document schema are detailed in our OpenDocument XML Citation Proposal.pdf

Implementing the new citation element in *xmloff* (the XmlOffice module) is a routine task. The Sun developers want to do it together with our programmer, so that he/she can learn how *xmloff* works.

The changes to the document schema need to be supported by the document save and load modules. The API module(s) concerned are:

interface XComponentLoader "This is a simple interface to load components by an URL into a frame environment."

which supports *loadComponentFromURL* and *storeAsURL*

See the Development Guide explanation for - 6.1.5 Handling Documents

??? is this correct and what else.

2. Modify the writer code to insert and display the new format citations

The bibliographic modules in OOo Writer need to modified to support the new schema. The modules that need to be modified are

- Bibliography
- textfield/Bibliography
- FieldMaster/Bibliography
- BibliographyDataField

3. Add support in the OOo save file package for storage of document bibliographic data.

Currently the Writer saves a complete copy of the bibliographic data associated with a citation, with each ciation. We propose to separate the citation and the bibliographic data, by leaving just the citation details in the document save file and place the detailed bibliographic data in a seperate bibliographic data file the OOo save file package.

The task is to complete the design of the bibliographic data file and add support for it in the OOo save file package.There is description of the XML Package, and is a FAQ about it.

4. Modify the Writer save-file read and save modules to support the new the bibliographic data file in the document save package.

The relevant component is "interface XComponentLoader" which supports *loadComponentFromURL* and *storeAsURL*.

5. Refine and Improve the CITEPROC bibliographic formating engine.

Experience XSLT programers are needed to work on this core component of the Bibliographic facility. It is functioning and a book has been published which used it to format the bibliographic table and citations.

We propose to build Bibliographic table and citation formating using XSLT style-sheets with a process called CiteProc. Also see BiblioX for technical discusion of this approach.

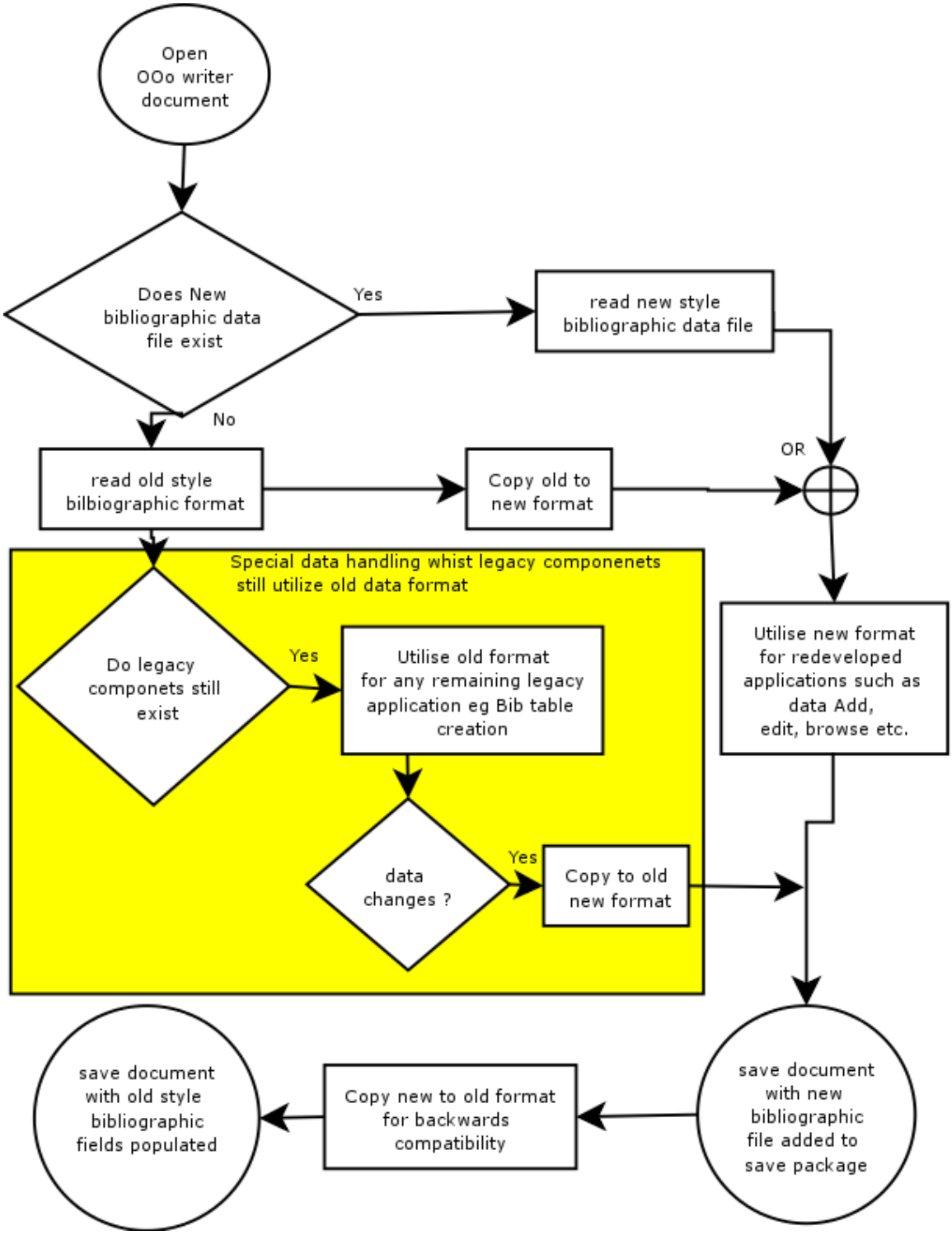
2nd Stage, Bibliographic Facility Redevelopment

top of page

1. Add Backwards and Forwards Compatability Logic to Writer

An important object of Bibliographic Enhancement project is to maintain document file backwards compatibility with older versions of OpenOffice. To achieve this when Bibliographic Entries are inserted into a Document they are stored with the same format as is currently the case. A new bibliographic entry tag will be will be added with the enhanced citation functions. Also a copy of the bibliographic data will be saved in the document save package. Older version of OpenOffice will read the old format of the bibliographic citations and ignore the bibliographic data file in the save package. The proposed enhanced OpenOffice will function as illustrated below

When a major revision of the save package format is introduced the support of the older bibliographic representations can be dropped.



The API module(s) concerned are:

interface XComponentLoader "This is a simple interface to load components by an URL into a frame environment."

which supports *loadComponentFromURL* and *storeAsURL*

See the Development Guide explanation for - 6.1.5 Handling Documents

2. Add Z39.50 and SRU/W support for the Bibliographic modules.

Build Z39.50 and SRU/W based internet searching facility using the YAZ toolkit (C & C++). This would enable searching for and retrieving bibliographic data from internet sources and storing them in a document or bibliographic database.

There is also a demonstration client program - IRTCL (requires YAZ and Tcl/Tk libraries be installed) that can perform the reference searches. It does everything but save or export the results ! However it is good model of how to use the toolkit and could be used as the basis for or model of a prototype internet searching facility. Screen pic, screen pic2.

A demonstration internet searching facility that writes selected bibliographic records back to the OOo bibliographic database has been written in Python - PyOOBib. Also instructions are available. Various problems with OOo Python have lead to us concluding that YAZ would be a better foundation than the Python code.

Also build Z39.50 and SRU/W server capability into OOo to enable users to share their bibliographic (and other) databases over the internet. One of the Indexdata toolkits could probably used as a basis.

The modules that may need to be modified are:

- Bibliography
- textfield/Bibliography
- FieldMaster/Bibliography
- BibliographyDataField

NB: We are considering using SWU/W as the standard method for OOo retrieving bibliographic data from any source. So that even a local Bibliographic database would also be accessed through SWU/W methods. The user would just select a local or remote source and the same access mechanism would be used.

3. Design and Build a basic Graphical User Interface (GUI). To provide -

- Basic citation insertion
- Basic bibliographic data entry
- Citation and bibliographic table formating using Citeproc.
- Basic Bibliographic database access
- Basic bibliographic internet search and database storage.

How to get started

top of page

Access to the source code for this project is available for download via CVS. A child work space has been created for us called "metabib" which contains a copy of the *xmloff*(OpenOffice.org XML File Format Definition) and sw (the word processor application component and the WYSIWYG HTML editor component) code.

The down load size will be about 1GB(?). And you will need about 2GB of disk space to compile the metabib CWS (Child-Work-Space). (Web access to CWS). If you can not handle that size download then ask us about sending it to you on cdroms.

Administration process - you first need to sign the JCA and then obtain the ssh key. After that we will show you how you can access the 'CWS'. It's basically a CVS branch. The most complicated thing is the setup of your tools, such that you can participate in the OOo development --- but, when you have got the ssh key we will show you.

See OpenOffice.org For Developers for general development information.

Sample Code

top of page

- Sample python code that reads and outputs some of the fields of the records in the bibliographic database. biblioacess.py
- Sample OpenOffice Basic program to write records to the bibliographic database bibwrite.html
- Henrik Just's LaTeX and BibTeX export filter http://www.hj-gym.dk/~hj/writer2latex/
- Applications which interact with Openoffice- Bibus (WxPython) and B3 (Java).
- A Perl module OpenOffice::OODoc provides a simple way to access document elements in the (closed i.e. not interactive with OOo) document save file. An example which retrieves bibliographic details is provided.

Contacts

Question or comments can be put to the Bibliographic Project development list dev@bibliographic.openoffice.org or to the project co-leader David Wilson.
