

KOffice: The Road Ahead

7. Juni 2004

Rechtlicher Hinweis

Dieser Beitrag ist lizenziert unter der Creative Commons License.

Zusammenfassung

While the KDE Project has successfully provided a compelling, feature-rich, and attractive desktop environment for either home or corporate users, there is still a big need for an integrated and user-friendly office suite. Since its beginning of development in 1998, KOffice tries to fill this gap, by providing a bundle of applications designed for daily office needs. This talk will try to highlight and live-demo key features of each application bundled in KOffice, covering KWord (word processor), KSpread (spreadsheet) with integrated KChart (charting component), KPresenter (slide presentation), Kivio (flowchart and business diagram), Karbon14 (vector drawing), Kugar (reporting tool), and Kexi (database). The talk will also describe the technologies used by KOffice, such as its XML document format, KParts-based component technology, advanced storage and printing system, rich text component for word processing and presentation, as well as filters for importing and exporting foreign file formats. Last but not least, will be discussed also the native support for OpenOffice.org/OASIS document format.

1 KOffice: The Road Ahead

The year of 2004 marks the 6th year of KOffice development, which was initiated and during the early stage of KDE (K Desktop Environment). With a goal of creating a polished, easy-to-use, integrated office-suite for Linux/Unix, KOffice has advanced from a test-bed for diverse KDE technologies into a pool of feature complete, fully functional applications for daily office needs: KWord for word processing, KSpread for spreadsheet, KChart for charting, KPresenter for presentation, Kivio for flowcharting, Karbon for vector-drawing, and Kexi for integrated database environment.

1.1 Technologies

A common feature in office suites is the ability to embed a document into another, for instance for a spreadsheet table inside a word process document. To achieve this, all applications must be available as components, and must be able to embed other components.

The component model used by KOffice is based on KParts. Components are made available as dynamically opened modules, which means KParts is an in-process component system. This gives better performance and better integration than out-of-process systems. KParts itself is a very simple model, with only one widget per component. KOffice, however, needs multiple views per document, and extends the KParts model accordingly.

A KOffice application defines a Document class, which holds the data for the document as well as the user's settings, and a View class, which presents the document to the user and handles user input. Qt's signal/slot mechanism makes it easy to connect the two together, removing the need for a Controller, as one can find in the common Model/View/Controller design.

A KOffice view is more elaborate than a usual widget, because it must support at least zooming and printing, so it must use real-world coordinates (like points or millimeters) as opposed to pixels. This is the reason simple widgets cannot be embedded into KOffice documents.

A view can be used as the main view in a toplevel window, but it can also be created when editing an embedded object. A particularity of the KOffice design is that an embedded object that is not being edited,

doesn't need to have its own view. For performance reasons and simplicity, embedded objects are being painted directly by their document. This is also used when creating the preview of a document (as used in the Konqueror file manager).

Printing with Qt is very easy. The same code as the one used to draw the data on screen, is used to draw it into a printer object, which handles the conversion to Postscript.

In order to provide full-featured rich-text editing to all KOffice applications, the text engine from KWord (originally from Qt) was moved into its own library, KoText, and is currently used by both KWord and KPresenter. The KOffice text engine provides many rich-text displaying and editing features, with styles, undo/redo ability, and WYSIWYG support for zooming and printing without unexpected reformatting of the document. The WYSIWYG support creates some problems currently with some fonts, but Qt4 will provide real builtin WYSIWYG support.

An important part of an office suite is the support for loading and saving files. Apart from the decision to use XML (since the very beginning of KOffice), an important part of the file format is how to store embedded objects and pictures. To this effect, KOffice uses a ZIP storage, just like OpenOffice.org does. A KOffice library, KoStore, provides a standardized API for loading and saving files in ZIP archives, tar.gz archives (for loading old files), and saving as plain files in a normal directory (as requested by some users for storing documents in CVS).

In the latest version of KOffice, 1.3, the XML being used by every KOffice application is specific to KOffice and had its share of inconsistencies and historical baggage. The goal for the next release is to switch to a standard file format for office suites. The first step was to participate in the specification of such a format, for which David Faure took part of the OASIS discussion group. The first version of the specification has been released in March this year, and the work of adding support for the OASIS format in KOffice is in progress.

The OASIS format was developed together with the OpenOffice.org developers, and the result is mostly based on the current OpenOffice.org file format, with extensions necessary for some KOffice features, including the Desktop Publishing support in KWord.

Office suites come with a plethora of import/export converters, also called filters. In KOffice, in order to keep the applications lean, the filters are separate modules, dynamically loaded. The KOffice libraries offer an architecture for writing and using filters. Most KOffice filters work by converting a file to a file (as opposed to loading a file into the application's data structures), which makes the filters available on the command line as well, by the means of the *koconverter* tool.

One of KOffice's main strong point is its integration with the rest of KDE. KOffice applications are DCOP-enabled which enables scripting and automation from the outside. KOffice uses KDE's printing architecture, kdeprint, which gives it support for many printing systems including CUPS, as well as the ability to print to a PS file or to a PDF file. KOffice is fully network-transparent, which allows users to work with remote files (over for instance FTP, SSH or SMB).

KOffice also saves previews of the document into the ZIP archive when saving which enables icon previews in Konqueror. For full-scale previews, KOffice also offers a read-only viewer which Konqueror can embed like any other KParts-based viewer.

1.2 KWord

KWord is a frame-based word processor. The original authors of KSpread were Reginald Stadlbauer and Torben Weis. Later on, it is further developed by David Faure, Thomas Zander, Shaheed Haque, and Laurent Montel. Currently KWord is maintained by David Faure.

Since it's designed for word processing application, obviously KWord uses the KOffice text engine (KoText) extensively.

KWord also has a frame-based design. Text boxes, pictures, embedded objects, formulas, table cells, headers and footers are all frames. Even the main text area of the page is a frame, or the multiple columns per page. Frames have many features (auto-resize, creating followup frames on new pages, creating multiple copies of the same content...) which are used to implement the various needs of the above-mentioned specific frames. The frame-based design gives KWord some DTP features, like preparing the layout of a magazine.

Users can insert pictures from a dozen of different pixel-based image formats, as well as vector formats including SVG and WMF. Encapsulated Postscript is supported, although as a pixel-based format.

Modern word processors have support for bidirectional text editing, as needed for languages such as Hebrew and Arabic. KWord builds upon Qt's bidirectional support for this, and supports both paragraph direc-

tions as well as mixing right-to-left with left-to-right runs of text in the same paragraph.

Among other KOffice applications, perhaps KWord is the one which has the most complete set of import and export filters. As of now, KWord can support the following document format: Microsoft Word (import only), OpenOffice.org Writer, WordPerfect, Abiword, Microsoft Write, Rich Text Format (RTF), StarWriter, Ami-Pro, HTML, WML, and of course simple plain text.

The notably improved filter is Microsoft Word filter, which is based on libwv2, a library for reading and parsing Microsoft Word files. Beside its stability, this new libwv2-powered filter already offers support for importing the widely used document features of Microsoft Word, including text formatting, fields, header/footer/footnotes, bullets and numbered paragraphs, tables. Support for importing images is under development.

As stated before, KOffice applications plan to adopt the OASIS standard as the native document format. KWord is at the avant garde at the moment. Based on the filter for importing OpenOffice.org Writer documents, the native support for OASIS XML format is currently being added to KWord. It is expected that the next stable version of KWord is already feature-complete with respect to OASIS document support.

Another improved filter which is not of least importance is Rich Text Format (RTF). Since writing an export filter for the Microsoft Word .doc format is proved to be a very demanding task, RTF is being used instead, as a good "bridge between KWord and Microsoft Word. The end result is the same, since the RTF format contains all the necessary features.

One kind of unique filter which KWord has the PDF import filter. This allows KWord users to edit text and image from a PDF file, something which can be really useful when the only source you have for a document is a PDF.

1.3 KSpread

KSpread is a spreadsheet application. There are not so many similar applications for Unix, KSpread is among the ones which are still actively maintained and developed. The original author of KSpread was Torben Weis, a core KDE developer during its early years. Ever since there were few developers who worked to improve KSpread, among others David Faure, Laurent Montel, Phillipp Mueller, Ariya Hidayat, Norbert Andres, and Lukáš Tinkl. Currently KSpread is maintained by Ariya Hidayat and Laurent Montel.

To implement spreadsheet functionalities properly, KSpread has its own version of the so-called spreadsheet object model which follows the common hierarchical model of spreadsheet: Document, Workbook, Sheet, Cell. Quite a number of KSpread features are implemented around the object model. For example, various possibilities of cell formatting is simply a matter of changing the formatting property of the cell. There are various on-going attempts to simplify the data structure used to hold the spreadsheet object model, so it is expected that in the next release, KSpread would become much more efficient with respect to memory usage.

Being able to calculate 'formula' is perhaps the heart of spreadsheet application. For this purpose, KSpread uses a library called KoScript. Originated from KScript (which was designed as KDE scripting engine but never really took off), KoScript parses and evaluates formula, even with supports for run-time functions. At the moment, KSpread provides more than 300 assorted functions, from mathematical, text manipulation, conversion, financial, and statistics. A new replacement for KoScript is being worked on. Unlike KoScript which analyzes expressions and creates the appropriate parsing-tree, the new formula engine uses another method: bytecode compilation. The lexical analysis part of this formula engine also offers support for translated function names.

Beside its native XML format, KSpread also supports importing and exporting other types of spreadsheet documents. At the moment, KSpread can handle Microsoft Excel (only import), OpenOffice.org Calc (import and export), Gnumeric (import and export), Quattro Pro (only import), as well as simple CSV (Comma Separated Value). In the spirit of moving to OASIS XML document format, the development version of KSpread already has preliminary support for loading and saving to this format, based on the working implementation of OpenOffice.org Calc filter. Furthermore, there is work to rewrite the Microsoft Excel filter system, not only to improve support for importing Microsoft Excel workbooks, but also to provide possibility to save the document as Microsoft Excel .XLS file, one of the long-requested features.

Being based on KParts, KSpread is able to embed another KOffice part. A striking example of such embedding use is the charting support in KSpread. For this, KSpread relies on KChart, a generic chart component for KOffice. Although it can be used as stand-alone application to create assorted types of charts, KChart is normally used together with KSpread. In practice, making a chart is as simple as selecting the range of the data, activating KChart, and choosing the desired chart type. Of course, further customization is always possible.

At the heart of KChart is KDChart, a charting engine contributed by Klarälvdalens Datakonsult which is powerful and highly configurable. KDChart, and thus KChart, supports many different chart types (with proper 3-D view if specified): bar, line, area, high-low, ring, and polar, with customizable chart axes as well as header and footer.

1.4 KPresenter

KPresenter is an application to create screen presentations. KPresenter's original author was Reginal Staldbauer. Since he left the development team, KPresenter was further developed by David Faure, Laurent Montel, Werner Trobin, Toshitaka Fujioka, Lukáš Tinkl, Thorsten Zachmann, and Percy Leonhardt. Current KPresenter's maintainers are Lukáš Tinkl and Laurent Montel.

As for any other presentation program, KPresenter uses the concept of 'slide' or 'page'. During the slide show, each of the slides will be shown one by one, in full-screen. KPresenter has few templates which can be used to quickly prototype the slides. Since presentation slides typically consist not only of texts but also pictures or diagrams, KPresenter also offers drawing tools, covering from simple straight line to different kind of autoforms. Objects in the slides can be formatted, aligned, or just grouped together. One can also make certain objects 'sticky', i.e. they will eventually appear in all slides. In addition to this, there are around 40 slide transition effects which can be used to make the presentation more attractive.

When the presentation must be posted to the web, KPresenter's HTML slide show can be handy. Basically what it does is creating an image representation of every slide, put together in HTML pages, along with one index file, all of which are ready to be published.

Unlike KWord and KSpread, the filters for KPresenter didn't evolve too much in the last few years. Aside from a very basic Microsoft PowerPoint import filter, the best filter which KPresenter has at the moment is fully-functional OpenOffice.org Impress import and export filter. There is also an import filter for MagicPoint presentation which itself could be considered particularly special since it's written in Python. It is also a proof that developing filter for KOffice application doesn't require depth knowledge of C++.

It is interesting to note that KPresenter is often used by KDE developers when they deliver presentations in conferences. The above mentioned web presentation feature is also used to make the presentations available for other Internet users. This 'dogfood' philosophy is proved to be useful to track bugs (because developers would be more aware of bugs when they experience themselves), and also to show that KPresenter is already stable and usable enough for such purposes.

Just like KWord and KSpread, there is on-going work to implement native support for OASIS document format. Again, this is based on the working OpenOffice.org Impress import and export filter. Users of the next forthcoming release KPresenter should be already enjoy this feature. Some other improvements are also planned, although not being worked on yet. These will cover e.g. masters for slide and title, separating content and slide layout, usability improvement, more professional-looking templates, more complete Microsoft PowerPoint filter, and many more.

1.5 Kivio

Kivio is a flowcharting and diagram application. Initially developed by Dave Marotti, Kivio is now being maintained by Peter Simonsson, with help from Ian Reinhart Geiser, Laurent Montel and Frauke Oster.

From the first impression, Kivio is clearly KOffice's answer to Microsoft Visio. Using the familiar interface just like Visio, Kivio also has the same drawing concept: it uses stencils to help the user quickly create the diagram. Basically, there are predefined shapes in each stencil; those shapes can be easily dragged into the document and hence reduce the need to manually draw them. Of course, once the shapes from the stencil become objects in the document, they can be formatted, colored, aligned, and so on. Just like Visio, there is also the possibility to connect one shape to another shape, a feature which is not seldom used in daily drawing. Text, again with many formatting properties, can be placed for annotation.

Kivio comes with a handful of useful stencil sets, from basic flowchart shapes, UML blocks, flags, to much more complicated network hardware symbols. Aside from its own stencils, Kivio can also import and shapes from Dia.

Unlike other KOffice applications, Kivio doesn't support any other drawing type except its own native XML format. Since Kivio must face Microsoft Visio head-to-head, the ability to understand Visio's VSD document would be a great advantage. Unfortunately, the format specification of VSD is not open and kept secret by

Microsoft so the only way to build the filter is by reverse-engineering. Up to now, it seems nobody felt bold enough to undertake this delicate job.

Kivio is still actively developed and will enjoy some more addition of stencil sets in the future, like what has happened in the past. It is also planned to develop an autorouting connector which is quite necessary to support complex flowcharts or diagrams. Text support would be also improved with some rich-text formatting, instead of current basic text feature.

1.6 Karbon

Karbon, or more officially Karbon14, is a vector-drawing application. Its original authors are Rob Buis and Lenny Kudling. Lately Benoit Vautrin and Tomislav Lukman joined the development team.

Before Karbon14 became part of KOffice, there was another vector drawing application called Kontour, formerly known as Killustrator. Later on the development of Kontour was suspended, but fortunately Karbon14 started to be usable and even moved already into KOffice.

Karbon offers a huge number of features. As with any other similar application, it is possible to draw different kind of lines, curves, polygons, shapes and texts, all of them being path based. Some nice effects can be performed using this path system, for example text can follow a certain curve, not merely vertically or horizontally straight. Colors can be specified in either RGB or CMYK. There is also support for gradient fill and opacity. Even shapes can be manipulated, using intersection and addition. Several plugins are available for further object manipulation: path flattening, knots insertion, corners rounding, shadow and whirl/pinch.

To handle drawings with many many objects, layers can be used to group these objects logically. Obviously standard object tools, including alignment and grouping/ungroupings are one-click away. Using a Libart-based painter system, the rendering will be anti-aliased, making the drawing look smooth and less jaggy. In some cases, the wireframe view is useful to speed up the rendering. Karbon supports zooming in and out some parts, especially needed when working with complex drawings.

Vector drawing programs sometimes can be difficult to handle. Taking into account such problems which might be faced by the users, Karbon always shows a handy, short context-sensitive help text (can be turned off of course). There is also a bird-eye view docking window with preview for easy navigation.

As for import and export filters, at the moment Karbon offers support for various drawing format, including for example Windows Metafile (WMF), Adobe Illustrator (AI), PostScript (PS), and Scalable Vector Graphics (SVG). The latter is quite important since it is more and more adopted as de facto vector format for web applications. Being able to import WMF graphics is also useful in case of migration, considering the fact that WMF is widely used in Microsoft Windows platform. When the drawing must be exported as bitmap image, Karbon provides the possibility to save it as in PNG (Portable Network Graphics) format. This makes Karbon useful to design and draw icons.

Since Karbon supports embedding other Koffice components, it is really easy to insert a drawing inside text, spreadsheet, or presentation document (imagine putting the company logo within the annual financial report). In fact, this is possibly one big advantage of Karbon: it is easy to use and integrated to the rest of office suite.

1.7 Kexi

Kexi is an integrated environment for creating and managing databases. First developed by Lucijan Busch, Kexi now also enjoys the contributions from Peter Simonsson, Joseph Wenninger, Jaroslaw Staniek and Cedric Pasteur.

The motivation behind Kexi was due to the noticeable lack of a simple database application like Microsoft Access, FoxPro, or File Maker. All these are commercial, proprietary applications; the open-source community undoubtedly needs a similar application.

Kexi is developed with the rest of KOffice, although it was not released together with the rest in KOffice 1.3. At the moment, Kexi is already able to create database schemas, insert, query and process data. Kexi itself is not coupled with a specific database; instead such connection is provided via the database backend. At the moment the following database engine are supported: SQLite (file-based), PostgreSQL, MySQL and FireBird/Interbase. It is planned to offer support for other database engines as well.

Kexi's user interface resembles Microsoft Access. Working with certain databases means creating a Kexi project where all tables, queries, scripts and forms are stored. Since Kexi is just the development environment,

the actual data of course reside on the real database engine in use. One particular feature of Kexi that is really useful is the small docking window which displays context-sensitive help. This allows novice users to get familiar with Kexi environment, but could just become quick references for power users.

Kexi can be used both to create database tables or for data entry. In fact, making new tables or altering already existing tables could not be easier: just create or change the necessary fields and if necessary define the relationship among tables. No knowledge about the internals of the database engine, or even SQL, is necessary. Fresh new tables can be filled with data in a simple manner, i.e. using a spreadsheet-like form which supports auto-completion. Queries are constructed using fields drag-and-drop. Of course, normal SQL queries are still supported.

To facilitate user-friendly data entry, a form editor is being worked on. Basically, it is a tool similar to Qt Designer, to create user interfaces in a graphical way. The difference is that Kexi's form editor is tightly connected with the data source.

1.8 How to Help

KOffice always needs help. There are a few ways to contribute, which are described below. Details and more information can be found at <http://www.koffice.org/getinvolved/>

Contributing code is one of the biggest help KOffice may use. Of course familiarity of Qt and KDE programming is compulsory. For less difficult jobs, working on improving import and export filters would be very good as well. From time to time there are always easier and trivial tasks for fixing annoyances, possibly solved by a short patch only. As with any other open-source projects, development discussions happen on the koffice-devel mailing-list (<http://mail.kde.org/mailman/listinfo> <<http://mail.kde.org/mailman/listinfo>>) and IRC (irc.kde.org, #koffice).

Reporting bugs can be done using the bug tracking system at <http://bugs.kde.org>. This needs registration although it is pretty easy and not complicated. Bugs need to be reported as soon as possible, it is simply impossible for developers to be able to detect all the bugs and keep track of them individually themselves. Requests for new features can also be filed on the same site.

Translation is always useful. This ensures that KOffice would be available and useful to more and more people. The translation of KOffice is normally performed under the same guidance as the translation of the rest of KDE. Visit <http://i18n.kde.org> for details and more information.

Documentation of KOffice applications is another part which needs improvements and polish. Most of the time, what is written there is outdated, or not kept up to date with the development.

To make sure that new development would not break existing features or introduce fresh bugs, regression tests are needed. Although sometimes it is not always possible to test application functionalities automatically, having the test sets would allow developers to quickly pinpoint the existing problem and avoid unnecessary waste of time. This is especially true for import and export filters.

With the recent KDE Quality Teams Project, whose aim is to help people contribute to KDE, KOffice starts to get more exposure. This is not only necessary to increase the user base, but also very useful to get new contributors and resources. At the moment, the QA for KSpread is being worked on, but it is also expected that the same thing will cover other KOffice applications.

Donating is also a quite efficient way to support the KOffice project and the developers. This can be in the form of money, material or sponsoring. Please refer to <http://www.koffice.org/support/> for further information.

1.9 Conclusions

Not many people aware that KOffice was once mentioned in Microsoft antitrust trial (see http://linuxtoday.com/news_story-01-31-008-05-OP). By leveraging already proven and existing KDE technologies, and extending them to a certain degree, KOffice team already delivers a collection of applications satisfying the need for a lightweight, integrated office suite for home and small office. Given enough manpower and constant development pace, the bright future of KOffice is definitely sealed!