

GNOME Technologies in Real-World Applications

Federico Mena-Quintero

Ximian, Inc.

federico@ximian.com <http://www.ximian.com>

Abstract

One of the goals of the GNOME project is to address the lack of certain modern technologies in free systems. We have created a number of libraries that make it easier to write large-scale applications. This paper explains how the Evolution groupware suite uses the following features in GNOME: the Bonobo component model and the OAF object activation framework, the gnome-vfs library, the GNOME canvas display engine, the gnome-print libraries, the GtkHTML component, the Glade user interface builder and the libglade library, and how all of this is integrated together to create a good experience for the user.

1 Introduction

The GNOME project has created a good foundation for building graphical applications that are well-integrated within their desktop. Users have gotten used to features that Just Work(tm) in proprietary systems like Windows and MacOS: printing, advanced display capabilities, cut and paste, multi-part MIME mail, HTML displays, and a general sense of integration among it all.

The Evolution groupware suite (Evolution for short, or just Evo between friends) is one of the first applications that use most of the ‘new’ GNOME libraries; it is actually a real-world application that you may use every day, not just another CD player or IRC client that uses just the basic user interface libraries.¹ Evolution is a large application that makes use of a lot of the GNOME infrastructure. It is built out of components, so it uses the Bonobo component framework and the OAF daemon for object

¹Some people have written CD players with play lists that pull in a whole SQL framework. These do not count for the purposes of this paper.

activation. It represents file locations using URIs internally, leaving room for expansion. It needs printing capabilities, so it uses the gnome-print libraries. It needs sophisticated, flicker-free displays, so it uses the GNOME canvas extensively. It supports HTML mail and so uses the GtkHTML widget and editor component. Finally, it implements its own Camel library for handling mail.

2 Bonobo and Components

One of the goals of Evolution is to provide a set of reasonably fine-grained components that can be plugged together and used in other applications. Of course, we use the Bonobo component model to achieve this[2]. Evolution defines many interfaces using CORBA IDL and implements them based on the Bonobo conventions.

2.1 High-level User Interface

Five main components form the top level of the Evolution user interface: the shell, the executive summary, the mailer, the addressbook, and the calendar.

The shell is the main application window with the menu and tool bars, the shortcut bar, and the folder tree. When you select a folder in the shortcut bar or the folder tree the shell launches the appropriate component: if you click on the Inbox folder, the shell sees if the mailer component is running, launches it if needed, and then tells it to load the URI corresponding to the Inbox folder.

The components use Bonobo’s automatic menu and toolbar merging. The Evolution shell just provides basic menu and toolbar commands like File/Close, File/Exit, a few items in the View menu, and

Help/About. When a component gets activated it merges its own commands into the menu and toolbar. The mailer will add an Actions menu full of commands for manipulating messages and some commands to the View menu to let you specify whether you want to show/hide deleted messages, for example. The addressbook will add items to the View menu to let you select between a card-based view or a table view. The calendar has commands to select between daily, weekly, or monthly views. All the components have a File/Print command of their own.

2.2 How the Shell Launches Components

There is no special place to register the Evolution components that are installed on the system. Instead, the shell queries the OAF object activation daemon for servers that implement the `ShellComponent` interface. The query is as follows:

```
repo_ids.has ('IDL:GNOME/Evolution/
              ShellComponent:1.0')
```

That is, any object that implements the `Evolution::ShellComponent` interface is fit for consideration by the Evolution shell.

Objects that implement the `ShellComponent` interface have methods that the shell calls to create graphical views of a folder's data, as well as operations like creating and removing folders of the type that each particular component supports.

2.3 Interfaces of the Components

The different non-shell components in Evolution define their own interfaces. Some of them are for use only within Evolution, and some others are exported for the rest of the world to use.

The addressbook and the calendar both use a model/view separation between the data storage and the GUI views[7]. This means that the views do not manipulate or store the data directly, they rather display a representation of whatever the data model contains. The model also notifies the views when the data changes; the views do not know what

initiated the change in the data and they are just responsible for updating their graphical representations.

The addressbook and calendar data is managed by a separate component, the Wombat, that communicates with the GUI views which are implemented as Evolution shell components. The mailer does not use a model/view separation for various historical and technical reasons.

Other clients aside from the Evolution GUI views can make use of these interfaces and thus access the user's calendar and addressbook data. They do not need to know whether Evolution is running; any changes made to the data by these clients will instantly and automatically be reflected in the Evolution views if they happen to be active.

There are other miscellaneous components in Evolution such as the message composer and interfaces to create pluggable displays for the executive summary.

3 The gnome-vfs Library

Inspired by the Midnight Commander's virtual filesystem, the `gnome-vfs` library provides an asynchronous abstraction for a VFS[9]. It can access local files, FTP and HTTP sites, and other sources of data. In addition, it provides a convenient set of functions to manipulate URIs. Evolution uses the URI handling functions to store the locations of its mail, contacts, and calendar folders. It also uses the MIME-type facilities in `gnome-vfs` to handle different types of data.

4 The GNOME Canvas

Evolution uses the GNOME canvas display engine extensively. The `ETable` widget, a derivative of the canvas, was written so that Evolution could have a sophisticated display for tabular information. It has a model/view architecture and supports advanced features such as grouping based on columns, automatic sorting by multiple keys, and inline data entry.

The addressbook uses the canvas for its “minicard” view as well. To support automatic layout of the cards in columns the canvas was extended to have a reflow mechanism, EReflow. The reflow mechanism runs before the normal update cycle of the canvas; it asks the canvas items that have requested a delayed reflow to resize themselves and to recompute their layout.²

The calendar also uses the canvas for all its views. It is convenient to create custom canvas items for the background grid of the calendar views and then just put special text items on top of the grid as a representation of the appointments in the calendar.

5 The gnome-print Libraries

An important part of the GNOME framework is the gnome-print set of libraries and tools. These provide a Postscript-like imaging model for applications and a number of common dialog boxes that let the user select a printer, paper sizes, and other miscellaneous options.

The mailer, calendar, addressbook, and executive summary components in Evolution implement printing in different ways.

The mailer uses the printing functionality from the GtkHTML widget directly. Since mail messages get translated to HTML for rendering, Evolution can use this information for printing with the GtkHTML widget itself.

The calendar has a generic layout engine that is used to calculate the positions of appointments within the day and week/month views. These positions can also be used for printing, so the calendar has code that creates canvas items for display and that uses gnome-print calls for printing.

The addressbook has some very customized code for printing envelopes and address sheets. Also, the ETable widget can print its contents so the table view of an addressbook can print itself without using any special code in Evolution.

Finally, the executive summary renders itself using

²The GNOME canvas uses a delayed update/redraw mechanism that runs in the idle loop of the application. For a detailed description of this mechanism see [6].

HTML, so it can use the GtkHTML printing functionality in the same way as the mailer.

Applications that use gnome-print can automatically get a print preview window. Since the print preview and the printer drivers in gnome-print both use the Libart library for rendering, both will get the same nicely anti-aliased results.

6 The GtkHTML Component

GtkHTML is a port of the KHTML widget originally developed for the KDE project. It was ported to the GTK+ object system and the GDK and gnome-print backends for display and printing, respectively.

GtkHTML has considerable functionality for editing HTML in a graphical way, that is, without having to write markup yourself. The GtkHTML package includes a Bonobo control that can be embedded in applications that need HTML editing functionality. Like other Bonobo controls, it will automatically merge its menus and toolbar with those of the parent application. The mail composer window in Evolution uses the HTML editor control in this way; when the user finishes typing his message the composer wraps the HTML stream with the proper MIME and mail header chunks and spools it for delivery.

Future plans include using the HTML editor control for the calendar’s appointments and tasks. Right now Evolution only supports text-only descriptions for appointments, but making it use HTML instead would not be hard.

7 The Glade User Interface Builder

Being a complex application, Evolution has a large number of dialog boxes. Most of them are created using the Glade user interface builder[1]. Glade creates an XML file that describes the layout of the widgets in the dialog box. Evolution loads these files at runtime using the libglade library; libglade creates all the corresponding GTK+ widgets and displays them.

Using Glade provides the obvious advantage of not having to create dialog boxes by hand by writing code; instead you can simply lay out widgets graphically, give them names, and later reference them in your code. Also, the XML files that Glade creates can be localized easily by translators and they can test the results immediately without having to recompile the program; they can just see the translated dialog boxes in Glade itself or by restarting Evolution. Finally, the layout of dialog boxes can be changed without having to recompile the application at all as long as the widget names remain the same when you change the dialog box.

8 Conclusion

Evolution intends to provide a complete groupware application with all the features and polish that users have come to expect from similar applications in proprietary systems. The GNOME libraries provide the infrastructure that such a large-scale application requires.

References

- [1] Damon Chaplin, *The Glade User Interface Builder*; Ximian Technology white paper, 2000. <http://www.ximian.com/tech/glade.php3>
- [2] Miguel de Icaza, *Bonobo*; Ximian Technology white paper, 2000. <http://www.ximian.com/tech/bonobo.php3>
- [3] Miguel de Icaza, *The Gnumeric Spreadsheet: a Test-Bed for Component Programming*, Proceedings of Linux Expo 1999, <http://www.gnome.org/gnumeric>.
- [4] Raph Levien, *GtkCanvas and the Next Generation of User Interfaces*, Proceedings of Linux Expo 1999.
- [5] Michael Meeks, *The Bonobo Component Model*; Proceedings of the Ottawa Linux Symposium 2000.
- [6] Federico Mena Quintero and Raph Levien, *The GNOME Canvas: a Generic Engine for Structured Graphics*; Proceedings of the Usenix 2000 Technical Conference, Freenix track.
- [7] Federico Mena Quintero, *Evolution Calendar Framework*; Ximian Technology white paper, 2000. <http://www.ximian.com/tech/calendar.php3>
- [8] Ettore Perazzoli, *Evolution: The GNOME Groupware Suite*; Proceedings of the Ottawa Linux Symposium 2001.
- [9] Ettore Perazzoli, *The GNOME VFS library*; Ximian Technology white paper, 2000. <http://www.ximian.com/tech/gnome-vfs.php3>
- [10] Dan Winship, *Evolution*; Proceedings of the Ottawa Linux Symposium 2000.