How to Contribute to the GNOME Project

Paolo Borelli*, James Bowes†, Fernando Herrera de las Heras‡, Elijah Newren§ and Mariano Suárez-Alvarez¶
*GNOME Foundation, Casirate d'Adda (BG), Italy, pborelli@katamail.com
†Faculty of Computer Science, Dalhousie University, Halifax, Canada B3H 1W5, bowes@cs.dal.ca
‡Tecsidel S.A., Enrique Jardiel Poncela 6, Madrid, Spain, fherrera@onirica.com
§newren@gmail.com
¶msuarezalvarez@arnet.com.ar

Abstract—**This paper will cover the technical, social and bureaucratic parts of the GNOME development process, with the goal of allowing people who have knowledge about GNOME to contribute to, and help improve GNOME.**

Covered in this paper are useful resources for GNOME developers, including tutorials and application programming interface (API) documentation. Also outlined are typical tools used by developers, such as revision control systems. GNOME-Hello, a sample GNOME program, is also covered, to examine the typical layout of a GNOME program's source directory.

Many communication methods used by GNOME developers are covered as well. Mailing lists, Internet Relay Chat, and the World Wide Web are the three primary means of communication within the GNOME project. The World Wide Web provides a single-direction form of communication, usually providing the authoritative reference for GNOME policy. Mailing lists provide and archived, searchable method for communication. Internet Relay Chat provides a highly interactive, informal communication mode that is used as much for socializing as it is for GNOME development discussion.

I. Introduction

There has always been a strong demand at GUADEC for tutorials and guides on how to begin development with and for GNOME. To date, most of the focus at GUADEC has been on content for advanced developers. This paper will assist in filling this gap, based in part on the ongoing efforts of the Gnome-Love project [8].

This paper will show developers who are unfamiliar with GNOME how they can become contributors. It will cover the technical issues of the GNOME platform and internals, and address important details about the social and bureaucratic parts of the GNOME development process. It is the authors' intention to assist people in crossing the critical bridge from having knowledge about GNOME to applying that knowledge in a way to help improve GNOME.

II. Learn

GNOME is a large project; it provides a comprehensive desktop environment for the user, and so it is inevitable that the GNOME developer platform is just as comprehensive. Given this size, there are a large number of tools, libraries and technologies that a potential contributor must understand. Beyond the *technical* aspects of GNOME development, a contributor must also learn the *social* aspects of contributing to GNOME.

The learning curve of GNOME development, from a technical standpoint, is not as steep as it might first appear, after a few basics have been learned. Contributors are not are not even required to learn a new programming language. Granted, the core GNOME platform and desktop is written in C, but there are high-quality bindings available for a number of languages; desktop application authors can use any language of their choosing.

Many new contributors may feel tempted to start their own project as their first attempt at working on a GNOME-related program. Most authorities within the GNOME project will advise against it, however [6].

A. GNOME Technologies

GNOME provides a comprehensive suite of libraries and technologies for application development. This includes, but is not limited to:

- GLib – Lower level functionality to enhance the C programming experience. Many features of GLib, such as some of its data structures or string handling routines, may be included by default in the standard libraries of other programming languages. Thus, for these languages, GLib may not be as visible. GLib also provides the main event loop for GNOME programs, and GObject (which allows for object-oriented programming in C).
- GTK+ – The Gimp Toolkit. This is the graphical widget library used by GNOME applications. GTK+ provides the buttons, menu bars, and other interface elements which allow users to interact with GNOME programs.
- ATK – The Accessibility Toolkit. Applications and toolkits that support ATK can be accessed through alternative input and output methods, such as screen readers or magnifiers.
- GConf – Provides access to a configuration database. GConf provides a convenient method to save and load application configurations, and user preferences.
- GStreamer – An advanced multimedia framework. GStreamer allows applications to access and manipulate a variety of media types in a format and location-independent manner.

There are many excellent references, tutorials and books available for learning how to use these libraries and technologies [5], [3], [9].

B. Jhbuild and CVS

One of the first steps in contributing to GNOME is setting up a development environment. Such and environment includes, but is not limited to, compilers and interpreters for the contributor's chosen programming languages, development libraries, and tools that aid in application development (such as glade[1]). Most importantly, one must have a working copy of the program sources.

Development GNOME code is stored in the GNOME project's CVS[2] repository. Downloading and building GNOME sources from CVS manually can be quite complicated; most developers use a program called `jhbuild` which automates the process. In addition, `jhbuild` will also install the development code it downloads to a separate directory, leaving any existing GNOME installations untouched[3].

C. GNOME-Hello

Following a long-standing tradition in the hacker culture, the first program that one should look at when starting to learn about GNOME development is the GNOME variation of the classic "Hello, World!"; a simple program that displays this famous message.

GNOME-Hello can be checked out of GNOME CVS using the command:

```
export CVSROOT=:pserver:anonymous\
@anoncvs.gnome.org:/cvs/gnome
cvs login
cvs -z3 co gnome-hello
```

Alternatively, `jhbuild` can be used to download GNOME-Hello and all of its dependencies with the command:

```
jhbuild build gnome-hello
```

GNOME-Hello contains many files that are typically found in a GNOME program:

- `AUTHORS, COPYING, NEWS, README` – Informative text files containing information for users. Many GNOME programs also include a `HACKING` file, which contains any specifics for contributing to the program.
- `ChangeLog` – Every time a modification is made to the program sources, a note is written in this file. The note details who made the change, when they made it, and what the change was.
- `Makefile.am, configure.ac, autogen.sh` – These files are part of the program's build system. GNOME projects use Automake, AutoConf and Libtool to help automate building the program.
- `gnome-hello.desktop.in` – During the build process, `gnome-hello.desktop` is generated from this file. The desktop file follows the `freedesktop.org` standard [2] for describing a desktop application.
- `help/` – User documentation for the program, typically written in DocBook [1] format.
- `po/` – Translations for the program. GNOME uses gettext for internationalization [4].
- `src/` – The directory containing the program source files.

III. Get Involved

Open Source development and the Internet have always been tightly associated and GNOME is no exception; communication is at the core open source software. The whole concept of open source is about being able to share ideas, and the Internet makes this easy to do, allowing anyone to reach a word-wide audience.

This means that it is through the Internet that a potential contributor can find everything he or she needs to get started: tools, knowledge and other people who share the same goal.

The Internet isn't only the World Wide Web; it is also mailing lists, IRC, wikis and many other means of communication which allow one to be a part of the GNOME community. What follows is quick description of each one of these services, each one with its own advantages and disadvantages, and its own set of (usually unwritten) rules.

A. World Wide Web

Arguably the most familiar of all Internet technologies, the World Wide Web provides a gateway into the GNOME community. The official GNOME website can be reached at `http://www.gnome.org` and provides a large set of services and information. Of particular interest to a contributor are:

- `http://developer.gnome.org` – Hosted here are all of the official programming documents and API references. Also of great interest are white-papers, Human Interface Guidelines and the homepages of many subprojects like internationalization, documentation, accessibility and usability.
- `http://cvs.gnome.org` – Hosts viewcvs, bonsai and lxr. These are tools which allow one to see the source code hosted on gnome cvs directly from a browser, and perform queries against it.
- `http://planet.gnome.org` – A weblog aggregator for the GNOME community. While it isn't an 'official' publication, it is definitely a great way to find out what is going on in GNOME land, both technically and socially.
- `http://foundation.gnome.org` – The GNOME Foundation's home page. The GNOME Foundation is a non-profit organization whose goal is to further the work of GNOME.

[1] A rapid application development tool; glade allows for point-and-click user interface development.

[2] The Concurrent Versions System, available from `http://www.cvshome.org`. Other revision control systems popular among GNOME developers include Subversion (`http://subversion.tigris.org/`), and GNU Arch, which is available as the `tla` (`http://www.gnu.org/software/gnu-arch/`) or `baz` (`http://bazaar.canonical.com/`) implementations.

[3] This is important; development code can be quite unstable (prone to crashes, for example), and should not be relied upon for normal usage.

[4] The GNOME Translation Project, `http://developer.gnome.org/projects/gtp/`, hosts a number of excellent resources for translating programs.

There are many other sites which are useful for anyone who wants to get involved in gnome:

- `http://gtk.org` – GTK+ is the toolkit at the base of GNOME.
- `http://freedesktop.org` – Hosts a number of software projects and standards, all with the common goals of improving interoperability and advancing the state of the X desktop. Many components of GNOME are based on freedesktop standards.
- `http://gnomefiles.org` – A 'GTK+ Software Repository'. One can find nearly all GNOME-related projects listed here.

B. Mailing Lists

Mailing lists provide archival backup of all discussion which takes place on them. Because of this, many discussions regarding decisions on policy or the direction of GNOME will occur on mailing lists.

The GNOME mailing lists are hosted on `mail.gnome.org`. From `http://mail.gnome.org`, one can subscribe to GNOME mailing lists, modify existing subscriptions, or search through the mailing list archives.

Some important lists that all GNOME developers may want to subscribe to include:

- `desktop-devel-list` – The list that no GNOME hacker can do without. The Desktop development list plays host to announcements about experimental new features, discussion about the future of GNOME software modules, and matters of development policy. It should be noted, however, that this list is not for general discussion; one should ensure that any posts made to this list are necessary, and informative.
- `gnome-announce-list` – New software releases are announced on this list. Following it is a good way to keep track of new advancements in the GNOME desktop and related programs.
- `foundation-list` – The general discussion mailing list for the GNOME Foundation. This list is a good place to talk about advocating the use of GNOME, or matters of foundation policy.
- `gnome-love` – A list for new developers to get help, and old developers to give help. The theme of the GNOME Love list is helping new developers get involved with GNOME, but off-topic posts are not discouraged.

Contributors may also wish to subscribe to the mailing list relating to the specific program they are working on, if it exists, or to other broad-topic lists, like `usability-list` (for discussion regarding human-computer interaction topics in GNOME).

C. Internet Relay Chat

Internet Relay Chat (IRC) is commonly used within the GNOME project. IRC lets developers communicate instantly, without the lag associated with e-mail. Typically, IRC has a much more relaxed atmosphere than e-mail, and so IRC is used as often for socializing as it is for facilitating work within the GNOME project.

IRC groups its users into *channels* (like chat rooms), where each channel can have its own theme, and a user can join an arbitrary number of channels. Most GNOME and GNOME-related IRC channels can be found on the GIMPNet IRC network[5] (accessible through the `irc.gnome.org` server, for example). A good IRC client for the GNOME desktop is XChat[6]. Some important IRC channels for GNOME contributors include:

- `#gnome` – A channel for general discussion about GNOME and technical support for using GNOME. It is mainly for users.
- `#gnome-hackers` – Most GNOME developers use this channel, either for technical discussion, or just chatting.
- `#gnome-love` – Come here if you need any help with development, or beginning to develop GNOME and GNOME applications.
- `#bugs` – Discussion related to bug fixing and bug triage using GNOME's bugzilla installation.

Like the mailing lists, there are also IRC channels for specific programs, user groups (such as #gnome-fr, for French-speaking GNOME users), or language bindings. A list of all IRC channels can be seen by typing `/list` into an IRC client when connected to a server (this list can be quite long).

D. Wiki

Wikis have recently been gaining popularity as collaborative workspaces among Internet users. A wiki is a content management system that allows anyone to easily alter the content on existing pages, or to create new pages.

The GNOME project uses the *GNOME Live* wiki, located at `http://live.gnome.org`. Most of the content on GNOME Live is geared towards developers; GNOME Live is not really geared towards use by general GNOME users.

Like the mailing lists and IRC, GNOME's wiki has workspaces for various GNOME programs and work groups.

E. Learn the Culture

A particularly attractive feature of the GNOME project, like many open-source software projects, is its strong community. Typically, one can learn a lot by 'lurking', or simply watching what others do, before actively participating. By subscribing to and reading some of the GNOME mailing lists, reading Planet GNOME, and following the conversation in some GNOME IRC channels, a contributor can get a sense of what is acceptable within the GNOME community.

Many essays are available on the topic of integrating ones self into the open-source culture [7], [6].

[5]Other channels important to GNOME developers (#gstreamer, for instance) are on the Freenode network. See `http://www.freenode.net` for more information on Freenode.

[6]XChat is available from `http://www.xchat.org`. Another option is xchat-gnome, a front-end for XChat that aims to integrate better with the GNOME desktop. It is available from `http://xchat-gnome.navi.cx/`.

1) A Note on Etiquette: Most people working on GNOME are doing so on their own time, and out of a love for the project. This should be acknowledged and respected; Most contributors to the GNOME project have limited time, and want to use it as effectively as possible. Many of the GNOME mailing lists discourage posts that do not relate to the designated topic of the list. Similarly, *flaming* and *trolling* (posting intentionally insulting and hostile messages) are almost universally discouraged within the GNOME community.

IV. Find Tasks

It is very common in the open source community to hear that a program or a feature has been developed "to scratch an itch" by the programmer. In other words, many features are written to satisfy a need felt by the programmer herself. Doing this is good because the developer has an high interest in implementing the feature and knows exactly how he or she wants the program to behave.

This is not always the case, however, so the question that arises is: where can a potential contributor find a list of ideas to implement or of bugs to be fixed? The answer is not universal, but in the GNOME project the first place to look is GNOME bugzilla, at `http://bugzilla.gnome.org`. Bugzilla is a web based bug tracking system where most of the GNOME projects keep a list of bugs and requests for enhancements.

A very useful feature which bugzilla provides is the ability to compile reports for bugs which match a particular set of requirements. Of particular interest to a new contributor is the *gnome-love* report, located at `http://bugzilla.gnome.org/reports/gnome-love.cgi`, which lists all the bugs that each module maintainer has marked with the *gnome-love* keyword. The keyword indicates that the bug can be solved without knowing all the inner details of the module's codebase, and that the maintainer would appreciate help on it, since they probably do not have the time to fix the bug herself. The bug report often contains suggestions on getting started in fixing the bug, as well.

Other common places to look for suggestion on things to implement are `TODO` files that many projects include in their source distribution, web pages on the project web site, and wiki pages, where many maintainers keep a road-map containing the features they plan to implement in the program. Failing these, a contributor may ask the module maintainers directly what kind of help they would appreciate, for instance by sending an e-mail to the appropriate mailing list.

Once a contributor has chosen what to work on, it is important that they get in touch with the maintainers of the module. This way, the contributor can make sure that the bug is still present in the latest version of the program, that the eventual new feature would be appreciated and that someone else is not already working on the problem.

V. Get Your Work Noticed

Once a contributor finishes the desired modifications to their copy of the code, he needs to submit them *upstream* (to the maintainer of the package) so the changes can be integrated into GNOME. The preferred way to do it is by creating a *patch*, which is a text file containing the differences between the modified copy of the source code and the original. Patches are created with the `diff(1)` program and can be applied with the `patch(1)` program. Nearly all GNOME module maintainers prefer the use of the *unified* format (obtained by using the `-u` option) for patches instead of the default format produced by `patch`. A more common way to create a patch, if the programmer has checked the source code out of cvs and then modified it, is by issuing the command `cvs diff -up > foo.patch` which outputs the differences between the local copy and the cvs repository into the text file `foo.patch`.

There are lots of suggestions on how to create a patch properly. It is strongly encouraged that one follow the same coding style used in the rest of the program, and to include a ChangeLog entry. This way, it is easier to understand what was changed and to properly credit each contributor. A more complete list of guidelines for creating a patch are at `http://live.gnome.org/GnomeLove/SubmittingPatches`.

Once the patch file is ready and tested it should be attached in bugzilla to the relevant bug report or, if one doesn't exist yet, to a newly created bugreport. Depending on the module, it may also be a good idea to send the patch to the relevant mailing list.

Usually patches are more than welcome by maintainers, but it may happen that they do not have time to review the patch in a timely manner. The contributor should not be discouraged by the lack of feedback from the maintainer and should from time to time remind the maintainer.

As the Pragmatic Programmers say, *Sign Your Work* [4]. A GNOME contributor should be proud of the work they have done, and not be embarrassed to take responsibility for the contribution that they have made.

Acknowledgements

The authors would like to thank Sebastien Bacher, James M. Cape, Davyd Madeley, Shaun McChance, Sriram Ramkrishna, and the rest of the `#gnome-love` idlers.

References

[1] "The docbook document type," January 2005. [Online]. Available: http://www.docbook.org/specs/cd-docbook-docbook-4.4.html

[2] P. Brown, J. Blanford, and O. Taylor, "Desktop entry specification." [Online]. Available: http://standards.freedesktop.org/desktop-entry-spec/latest/

[3] T. Gale, I. Main, and the GTK team, "Gtk+ 2.0 tutorial." [Online]. Available: http://www.gtk.org/tutorial/

[4] A. Hunt and D. Thomas, *The Pragmatic Programmer*. Addison-Wesley, 2000, ch. 8, pp. 258–259.

[5] E. Newren, "Developing with gnome," 2004. [Online]. Available: http://www.gnome.org/~newren/tutorials/developing-with-gnome/

[6] H. Pennington, "Working on free software." [Online]. Available: http://www106.pair.com/rhp/hacking.html

[7] E. S. Raymond, "How to become a hacker," 2001. [Online]. Available: http://www.catb.org/~esr/faqs/hacker-howto.html

[8] The Gnome-Love Project, "Gnome-love," wiki workspace. [Online]. Available: http://live.gnome.org/GnomeLove

[9] M. Warkus, *The Official GNOME 2 Developer's Guide*. No Starch Press, 2004.