GNOME and gedit development feedback

Sébastien Wilmet

March 16, 2022

Introduction

I wanted to give a feedback about GNOME and gedit development. Giving a feedback is important, especially from those — like me — who have left the project after having contributed a lot.

1 About the new gedit design

1.1 The HeaderBar

Several years ago, another contributor to gedit has done the work to switch gedit from a traditional design (a menubar and a toolbar) to the new design (with the headerbar containing the "sandwich menu").

Although I think the new design is suitable for some applications, in the case of gedit there are usability regressions that are quite problematic.

To give a single example: using the mouse for the undo/redo feature is now basically broken, you need to do right click → undo → right click → undo → ... With the toolbar, it was a single click.

Of course there are keyboard shortcuts to access features such as the undo/redo, but gedit primarily targets beginners (the next text editor to use after Notepad on Windows). So using the mouse is of primary importance, and keyboard shortcuts are for more advanced users.

1.2 Overlay scrollbars

In the past, scrollbars were always visible, but their design and colors were discreet.

Now, the GNOME designers want — at least for all core applications — overlay scrollbars only. When they appear, their new design and colors are made such that they attract the attention. It's easily possible to configure the scrollbars so that they are always visible, but then, it's seen as something "ugly". To bring back the more discreet colors, it is much more work for the application developer.

A simple switch to overlay scrollbars, in the case of gedit, introduced another usability regression when using the mouse: when you want to place the cursor at the last column or at the last line of text, the overlay scrollbar suddenly appears and you are unable to place the cursor where you want. The user needs to use a work-around

by placing the cursor (with the mouse) at a nearby location, and then move to the last column or line with the keyboard arrows (when using the keyboard only, the overlay scrollbars don't appear).

So what to do in such a situation, as a gedit developer? Prioritize design and the so-called "User Experience (UX)", or prioritize usability? In this case, I prefer something a little ugly but usable than something beautiful but that drives you nuts.

1.3 One principle of design and fashion

Also, I came aware that one principle of design and fashion is that it *needs* to change. If it doesn't change, the users are bored, it is less "marketing". But for the GNOME developers, it's a lot of work, sometimes some frustration and disagreements. Some users complain on IRC with sometimes insults, due to the design changes that degrades the usability.

2 Unclear about who decides what in GNOME

One problem, according to me, is that it's really not clear of who decides what in GNOME. According to certain persons, the people who contributes a lot have more influence on the decisions (with the experience, the respect, etc). Then, insidiously and over the (recent) years, it is the designers (with the Human Interface Guidelines to strictly follow) who went more or less above / the N+1 of the maintainers.

In addition, there is also the GNOME Foundation and the release team. Currently the release team decides on what applications are part of "GNOME Core", in other words, what the GNOME project recommends to the distributions for the default install.

In case of a disagreement, what to do? It is handled informally, but when more and more disagreements come up over the years, it creates more problematic controversies.

Note that in contrast to GNOME, the Debian project — for instance — has a much better, battle-tested way to handle social issues and disagreements among developers, packagers and other contributors.

3 Technical feedback about GNOME development

3.1 Lots of changes

These past years, there were lots of changes for GNOME developers:

- Flatpak.
- Meson.
- GitLab and Continuous Integration.
- BuildStream and gnome-build-meta.
- GTK 2 → GTK 3 → GTK 4 with many API breaks, without compatibility layers to ease the transition.

- Some so-called enhancements for writing GObject classes in C that cause more problems than it solve[1].

3.2 Too many possible programming languages

The GNOME community is not united.

Some developers prefer the C language, others prefer Vala, or C#, or C++, or Rust, or Python, or JavaScript, writing plugins in Lua, etc.

A developer might choose one language (e.g., C), and then several years later he or she wants to rewrite the code in another language (e.g., Vala), then there is a social pressure to rewrite everything in Rust. Where will it end?

4 A Free Software project is not an organization

The GNOME project is not an organization with workers bound by a contract. Note that there is the GNOME Foundation which is a non-profit organization, but the vast majority of contributors to the GNOME project (including companies) are not bound by a contract with the GNOME Foundation.

What does it mean? Individuals can come and go as they wish. Including of course maintainers of core GNOME components. And fortunately so, the contrary would be slavery.

So, when a core maintainer contributing his or her own time as an individual wants to do a pause, it doesn't fit well the GNOME schedule for releasing new stable versions every six months. Even worse so, other developers are eager to "take over" the maintenance and development of a module, taking sometimes a radically different direction for the project. As a result, when the first person wants to contribute again, after a well-deserved rest, he or she sometimes needs to fork the project to continue the development at the point where he or she left it!

Conclusion

In my opinion, heavily contributing to GNOME as an independent individual is ... complicated. I.e., not as an employee of a company that is well established within the GNOME community.

As you can imagine after reading the previous sections, there can be a lot of stress that GNOME developers endure.

I don't have good suggestions, apart from the companies around GNOME to hire more developers and other contributors and pay them well.

[1] The `G_DECLARE_FINAL_TYPE()` and `G_DECLARE_DERIVABLE_TYPE()` macros used in headers: switching between them involves a huge amount of work to adapt the `*.c` file.