Visualizing Gnome With The Small Project Observatory

Mircea Lungu, Jacopo Malnati, Michele Lanza
REVEAL @ Faculty of Informatics - University of Lugano, Switzerland

Abstract

We analyzed the Gnome family of systems with the Small Project Observatory, our online ecosystem visualization platform. We begin by briefly introducing the model of SPO. We then observe and discuss several phases in the activity of the Gnome ecosystem. We follow and look at how the contributors are distributed between writing source code and doing other activities such as internationalization. We end with a visual overview of the activity of more than 900 contributors in the 10 years of existence of Gnome.

1 Introduction

Software systems seldom exist by themselves. Often, they are part of a larger context, a veritable "software ecosystem", which is the collection the software projects developed within and across organizational boundaries, such as companies, research groups, and open source communities [1]. The projects in an ecosystem depend on one another, share code, share development methodologies and also developers: The developers in an ecosystem collaborate on various projects, work on projects in parallel or migrate between projects; they also depend on each other's code. To find such information one must look beyond the repositories of SCM (software configuration management) systems such as CVS and SVN, and step up one level to so-called "super-repositories". We define a super-repository as the collection of SCM repositories for multiple projects that belong to an ecosystem.

For the analysis in this paper we use the Small Project Observatory (SPO), our online ecosystem visualization tool. SPO is a web-based platform that supports the interactive exploration and visualization of software ecosystems [1]. Although SPO supports many types of analysis, in this article we focus our attention on the activity of the developers in the Gnome ecosystem.

[1]For more types of analysis and for the interactive versions of the visualizations presented here we invite the reader to visit the tool's website at `http://spo.inf.unisi.ch/`

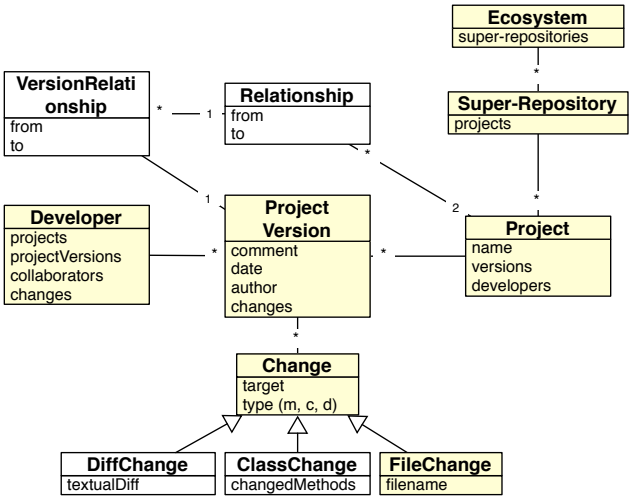

Figure 1. The ecosystem metamodel of SPO

Figure 1 presents the ecosystem metamodel that SPO implements. The central entity in our model is an *Ecosystem* which is comprised of multiple *Super-Repositories* which contain in turn one or more *Projects*. Every *Project* has *Project Versions* and every Project Version has an associated *Developer* and a set of *Changes*. *ProjectVersions* have *VersionDependencies* between themselves which are aggregated to the level of *Projects* as ProjectDependencies. In the context of this paper we focus on developers, and therefore we only use a subset of the ecosystems metamodel, denoted by the highlighted classes in Figure 1. We focus our analysis on the overall activity and trends within the Gnome ecosystem, the contributions of developers, and the history of their activity.

2 Analyzing Gnome with SPO

Overall Activity and Trends *How did the overall developer activity evolve in the ecosystem over time, and how do the individual projects contribute to it? Are there visible patterns in this activity evolution?*

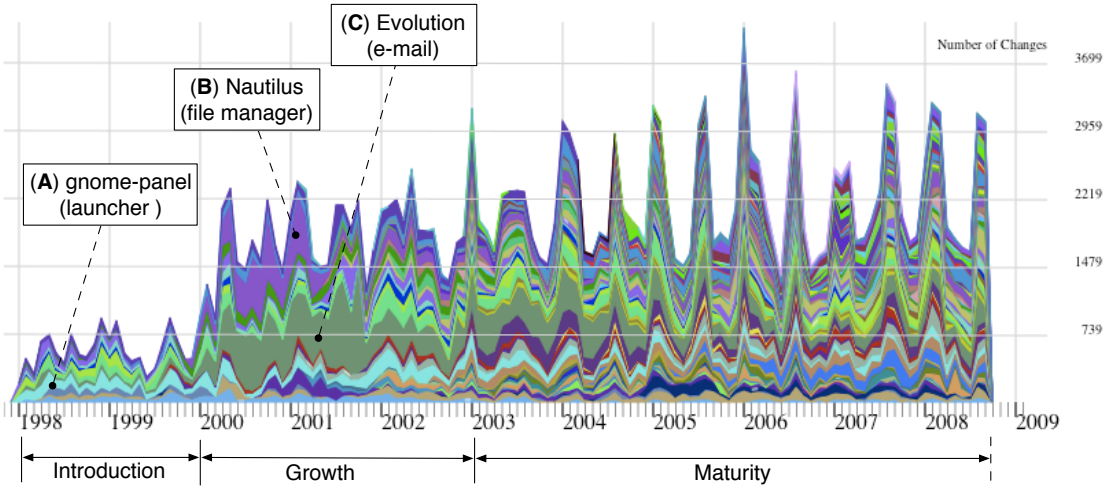

Figure 2. Visualizing 10 years of activity in the Gnome ecosystem

The visualization principle used in Figure 2 is to assign to each project a specific color, and represent it as a surface where the horizontal axis shows time and the height of the surface is given at every point by a metric (in this case the number of commits) computed at the respective point in time. The surfaces are stacked one on top of the other. We see both global trends at the ecosystem level and trends at the level of individual projects, distinguishing three phases of Gnome's lifetime:

1. *Introduction (1998 - 2000).* This period has few active projects with low overall activity. Some of the projects initiated here are still active at the time of writing of this paper (e.g., *gnome-panel*, marked as A).

2. *Growth (2000 - 2003).* The activity on two projects overshadows others. Marked with (B) and (C) in Figure 2 the *Nautilus* file manager and the *Evolution* e-mail client take at times the majority of the effort.

3. *Maturity (2003 - 2009).* There is no single project on which there is a focus in terms of activity but this period has a cyclical sequence of peaks and valleys of activity, pointing to development policies and release cycles. In every year there is a peak in January and July which sometimes doubles the number of commits in the previous month. Analyzing the commits for each peak we noticed that they concern both internationalization and source code files, but are not limited to copyright updates. Gnome ships each 6 months and these peaks represent the developer effort for the last month of each single release, as personally confirmed by some of them.

Individual Developer Contributions. *What is the distribution of effort between developers in terms of code versus internationalization?*

Gnome has a total of more than 900 developers that contributed both source code (in the form of `.c` and `.h` files) and internationalization support (in the form of `.po` files).

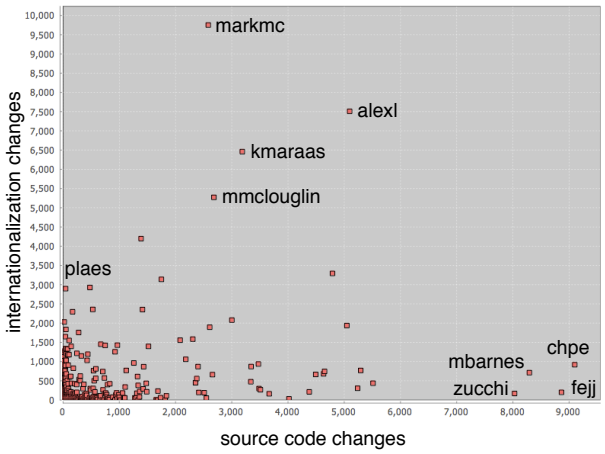

Figure 3. Developer effort distribution.

Figure 3 is a scatter plot presenting each developer as a red dot positioned according to the number of source files (horizontal axis) and internationalization files (vertical axis) that have been changed and committed. The majority of developers is in the bottom left corner, committing less than 1000 files, be it code or not. Many authors are contributing either source code or only internationalization files (*plaes*, for example, contributed about 3000 `po` internationalization

files and almost no source code). The whole scatterplot indicates a bias towards code-oriented development, according to the the numerous dots spread on the horizontal axis in the lower portion of the plot. Some outliers are clearly identifiable: Developers *chpe*, *fejj*, *zucchi* and *mbarnes*, in the bottom right corner of the plot, heavily contributed to the ecosystem with thousands of `c` and `h` files, while other developers, such as *kmaraas* and *alexl* committed roughly the same amount of source code and `po` files. We discovered that *mmclouglin* and *markmc* are two aliases for the same person who used them in two contiguous and separate periods. Both of these accounts have been used to commit thousands of source code and internationalization files, resulting in a new outlier that contributed about 15000 `po` files and 5500 `c` and `h` files. Figure 3 does not indicate which of the depicted developers is still active.

Figure 4. Tag cloud of the most active developers in terms of file changes

Figure 4 presents a tag cloud with the names of the most active developers in terms of number of file changes (code and internationalization). The size of each developer's name is proportional to the number of committed files, while their order is computed according to their first commit in the ecosystem. The color of the developers is assigned according to their weighted activity in the last 12 months. The darker the developer, the more active he has been. Among the previously discussed outliers just a few of them have been active in the last year: *fejj*, *alexl*, *kmaraas*, *mbarnes* and *chpe* while the others are light gray, denoting no or minimal activity. *fejj* and *alexl* appeared in the ecosystem in an early stage, which justifies the amount of their contributions, while developers *mbarnes*, *kmaraas* and *chpe* came later, contributing a huge amount of changes in a shorter time.

Developer Activity History. *How long do the individual developers remain active in the ecosystem? Are there developers who have been active since the beginning? Can we observe patterns of developer activity?*

Figure 5 presents the periods of time when each developer has been active in the ecosystem. Each row represents one developer, each column represents a period of one month. Each developer has associated a binary vector of activity which models whether he has been active or not in each month. The rows are arranged in such a way that developers that have similar activity patterns are clustered together using a hierarchical clustering algorithm in which the distance between two vectors is the Hamming distance between them. By traversing the resulting dendrogram we obtain an ordering of the vectors which keeps the vectors that are similar grouped together. When plotting the matrix, each developer is assigned a specific color. The figure reveals the following facts:

1. No developer was active from the beginning to the end. However, there are a few developers that were present for almost all the ecosystem's history. The developer who comes the closest to having been active throughout all Gnome's history is *krmarass*, with more than 100 commits every month since nearly the beginning of the project. The other developers that are active for long periods of time in the ecosystem are visible on the top of the graph in Figure 5.

2. Some contributors are just passing by. The bottom half of Figure 5 contains more than 450 developers who were active at a single point for a short period of time in the history of the ecosystem and then disappeared.

3. People come in groups. Figure 5 shows several clusters of developers who have similar patterns of activity in time. They arrive in the ecosystem about the same time and after a certain time they might leave together. Some developer groups have a short lifetime (e.g., C, D and E) while others have long lifetimes (e.g., A and B). Some of the clusters are the result of people working on the same project. For example cluster (A) is mainly composed of developers contributing to the Nautilus project.

3 Conclusions

Focusing our attention mostly on developers, we presented a series of observations about the Gnome family of systems obtained through the use of SPO, our ecosystem analysis platform.

References

[1] M. Lungu. Towards reverse engineering software ecosystems. In *Proceedings of ICSM 2008 (24th IEEE International Conference on Software Maintenance)*, pages 428–431, 2008.

Figure 5. The activity history of the more than 900 Gnome developers