Evidence for the Pareto principle in Open Source Software Activity

Mathieu Goeminne and Tom Mens
Institut d'Informatique, Faculté des Sciences
Université de Mons – UMONS
Mons, Belgium
{ mathieu.goeminne | tom.mens }@umons.ac.be

***Abstract*—Numerous empirical studies analyse evolving open source software (OSS) projects, and try to estimate the activity and effort in these projects. Most of these studies, however, only focus on a limited set of artefacts, being source code and defect data. In our research, we extend the analysis by also taking into account mailing list information. The main goal of this article is to find evidence for the Pareto principle in this context, by studying how the activity of developers and users involved in OSS projects is distributed: it appears that most of the activity is carried out by a small group of people. Following the GQM paradigm, we provide evidence for this principle. We selected a range of metrics used in economy to measure inequality in distribution of wealth, and adapted these metrics to assess how OSS project activity is distributed. Regardless of whether we analyse version repositories, bug trackers, or mailing lists, and for all three projects we studied, it turns out that the distribution of activity is highly imbalanced.**

***Index Terms*—software evolution, activity, software project, data mining, empirical study, open source software, GQM, Pareto**

I. Introduction

Numerous empirical studies aim to understand and model how open source software (OSS) evolves over time [1]. In order to gain a deeper understanding of this evolution, it is essential to study not only the software artefacts that evolve (e.g. source code, bug reports, and so on), but also their interplay with the different project members (mainly developers and users) that communicate (e.g., via mailing lists) and collaborate in order to construct and evolve the software.

In this article, we wish to understand how activity is spread over the different members of an OSS project, and how this activity distribution evolves over time. Our hypothesis is that the distribution of activity follows the Pareto principle, in the sense that there is a small group of key persons that carry out most of the activity, regardless of the type of considered activity. To verify this hypothesis, we carry out an empirical study based on the GQM paradigm [2]. We rely on concepts borrowed from econometrics (the use of measurement in economy), and apply them to the field of OSS evolution. In particular, we apply indices that have been introduced for measuring distribution (and inequality) of wealth, and use them to measure the distribution of activity in software development.

The remainder of this paper is structured as follows. Section II explains the methodology we followed and defines the metrics that we rely upon. Section III presents the experimental setup of our empirical study that we have carried out. Section IV presents the results of our analysis of activity distribution in three OSS projects. Section V discusses the evidence we found for the Pareto principle. Section VI presents related work, and Section VII concludes.

II. Methodology

A. GQM paradigm

To gain a deeper understanding of how OSS projects evolve, we follow the well-known **Goal-Question-Metric** (GQM) paradigm. Our main research **Goal** is to understand how activity is distributed over the different stakeholders (developers and users) involved in OSS projects. Once we have gained deeper insight in this issue, we will be able to exploit it to provide dedicated tool support to the OSS community, e.g., by helping newcomers to understand how the community is structured, by improving the way in which the community members communicate and collaborate, by trying to reduce the potential risk of the so-called *bus factor*[1], and so on.

To reach the aforementioned research goal, we raise the following research **Questions**:

1) Is there a core group of OSS project members (developers and/or users) that are significantly more active than the other members?
2) How does the distribution of activity within an OSS community evolve over time?
3) Is there an overlap between the different types of activity (e.g., committing, mailing, submitting and changing bug reports) the community members contribute to?
4) How does the distribution of activity vary across different OSS projects?

As a third step, we need to select appropriate **Metrics** that will enable us to provide a satisfactory answer to each of the above research questions. For our empirical study, we will make use of *basic metrics* to compute the activity of OSS project members, and *aggregate metrics* that allow us to compare these basic metric values across members (to understand how activity is distributed), over time (to understand how they

[1]The *bus factor* refers to the total number of key persons (involved in the project) that would, if they were to be hit by a bus, lead the project into serious problems

evolve), and across projects (to compare the situation between different OSS projects).

B. Basic metrics

To obtain the *basic metrics* of OSS activity, we will extract information from three different types of data sources we have at our disposal: version repositories, mailing lists, and bug trackers. For each of these data sources, we can define metrics that extract and reflect a particular type of activity:

- **Development activity**: the activity of developers committing source code to a version repository, measured as number of commits.
- **Mailing activity**: the activity of project members posting messages to a mailing list, measured as number of mails.
- **Bug tracker activity**: the activity of persons interacting with a bug tracker, measured in three different ways: number of new bug report submissions, number of comments added to existing bug reports, number of changes to existing bug reports.

Since we are not only interested in a static view of a particular snapshot of an OSS project at a particular moment in time, we will extract each of the above activity metrics during the entire life of the considered OSS projects.

C. Aggregate metrics

Since several of the research questions require a comparison of the basic metrics (across persons, across projects, and over time), we need *aggregate metrics* that combine the basic metrics. This is valuable, in particular, if we want to reason about the distribution of activity across OSS project members.

To study such distribution, we borrow ideas from *econometrics*. This discipline uses statistics and metrics to analyse economic data. As an example, various aggregation measures of statistical dispersion have been proposed (e.g., the Hoover, Gini, and Theil indices) and applied to assess the inequality of the wealth distribution among people, regions, countries, and so on.

Recently, some of these aggregation measures have been used for analysing evolving software systems. Vasa et al. [3] proposed to use the Gini index as an alternative to traditional software metrics. Serebrenik et al. [4] proposed to use the Theil index instead. Following this emerging trend, we will use three different aggregation measures to study OSS activity distribution. Below we provide the definitions of the three aggregation measures we selected: the Hoover index, the Gini index, and the Theil index. These definitions rely on two auxiliary definitions.

Let $X = \{x_1, \ldots, x_n\}$ be a set of values indexed in ascending order ($\forall i \in 1 \ldots n-1, x_i \leq x_{i+1}$). The sum of all these values will be called x_{total} (Equation 1). The mean of all values will be called $\bar{x}$ (Equation 2).

$$x_{total} = \sum_{i=1}^{n} x_i \tag{1}$$

$$\bar{x} = \frac{x_{total}}{n} \tag{2}$$

The Hoover index, defined in Equation 3, is one of the simplest ways to assess inequality of wealth or income. Its value is the ratio of incomes to take up from the richest part of the population to redistribute to the poorest one so that the incomes become perfectly equal. A Hoover index of 0 represents perfect equality, while a value of 1 represents perfect inequality.

$$H(X) = \frac{1}{2} \sum_{i=1}^{n} \left| \frac{x_i}{x_{total}} - \frac{1}{n} \right| \tag{3}$$

The Gini index, defined in Equation 4, provides a more complex (but also more representative) way to assess inequality of income. The cumulative function of income distribution is represented by a perfect diagonal if all entities in the population would have the same income, and by a curve under the perfect line otherwise. The Gini index is the value of the surface area between the perfect line and the curve, divided by the surface area below the perfect line.

$$G(X) = 1 - \frac{2}{n-1} \cdot \left(n - \frac{\sum_{i=1}^{n} i x_i}{\sum_{i=1}^{n} x_i} \right) \tag{4}$$

Yet another index to assess inequality of income or wealth is the Theil index. It is based on the Shannon entropy [5], and is defined in Equation 5. Because the Theil index is not bounded *a priori*, one cannot easily compare it with the two aforementioned indices. To normalize the Theil index so that it always returns a value between 0 and 1, we can apply the normalisation function N described in Equation 6 to it.

$$T(X) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i}{\bar{x}} \cdot \ln \left(\frac{x_i}{\bar{x}} \right) \right) \tag{5}$$

$$N : t \to 1 - e^{-t} \tag{6}$$

III. Experimental setup

A. Implementation

In order to obtain replicable and verifiable results, we do not only need a good methodology. At least as important is automated tool support that enables us to extract data from the different data sources, compute the basic and aggregate metrics based on this data, statistically analyse the obtained metrics, and visually confirm the results.

To this extent, we use and extend our generic framework for analysing open source software projects. This framework, presented in [6], has been developed in a modular and extensible way, facilitating the implementation of new modules meeting specific needs. Support for automatic extraction of data from version repositories, mailing lists and bug tracking systems was already built-in. The framework also supports generation and visualisation of various types of software (project) metrics.

For the specific purposes of this article, we added a new module to compute activity distribution (i.e. the relative activity of each involved person), and different variants of activity are supported. In particular, we implemented the three types of activity defined in Section II-B. We also added a module for computing aggregation indices. Among others, the Hoover, Gini, and Theil index are currently supported. The existing statistic analysis and visualisation modules can directly exploit the information computed by the activity distribution module and the aggregation index module to produce statistical output representing the inequality indices.

B. Pareto principle

Many types of distributions in which people are involved correspond to the so-called *Pareto principle*: roughly 80% of the effects stem from approximately 20% of the causes [7]. This principle and the associated law have been observed repeatedly in a variety of domains, including software evolution [8], [9].

Answering the first research question of Section II-A boils down to finding empirical evidence for the *Pareto principle* in OSS project activity distribution. One should note the difference between the Pareto principle and the related notion of Pareto distribution [10]. While a Pareto distribution satisfies the Pareto principle, the inverse is not true: a statistical distribution may satisfy the Pareto principle without being a Pareto distribution. In fact, many types of *power law* probability distributions have been observed when analysing human activity, and OSS project activity in particular, and the Pareto distribution is only one them [7], [11]. Many power law distributions satisfy the Pareto principle without being a Pareto distribution.

The second research question of Section II-A corresponds to determining whether the Pareto principle is present throughout the entire life of the project, and whether it emerges, stabilises or disappears over time.

C. Selected projects

We will analyse the distribution and evolution of activity on the following OSS projects: Brasero (projects.gnome.org/brasero), Wine (www.winehq.org), and Evince (projects.gnome.org/evince). They have been selected based on a variety of factors: popularity, age size, availability of the necessary data sources for analysis, and so on. Some of the characteristics of the three selected projects are presented in Table I.

IV. Empirical study

A. Brasero

Figure 1 shows the cumulative activity distribution in the Brasero community, for three types of activity: commits made to the version repository, mails sent to the mailing list, and changes made to bug reports in the bug tracker. These distributions are shown for the last version of Brasero we analysed, namely the one available on November 2010. For the previous versions, with the exception of the earliest versions, we get similar results.

OSS project	Brasero	Evince	Wine
main programming language	C	C/C++	C
versioning system	git	svn	git
age (in years)	8	11	11
size (in KLOC)	107	580	2001
# of commits	4100	4000	74500
# of mails	460	1800	14000
# bug reports	250	950	3300
# commiters	206	204	1229
# mailers	102	610	6879
# bug reporters	386	961	2676

TABLE I
MAIN CHARACTERISTICS OF SELECTED OSS PROJECTS. THE REPORTED VALUES HAVE BEEN COMPUTED FOR THE LAST VERSION, NOVEMBER 2010.

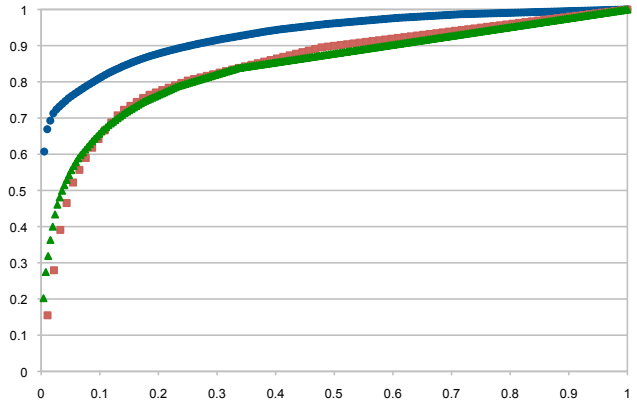

Fig. 1. Cumulative view of distribution of activity for Brasero (November 2010). The x-axis shows the cumulative percentage of active persons, ordered from most to least active; the y-axis shows the cumulative percentage of activity. The upper distribution (blue circles) corresponds to the commit activity . The distribution with red squares corresponds to the mail activity. The distribution with green triangles corresponds to the bug report change activity.

These distributions illustrate that there always is a small core team of persons that account for most of the activity. For the commit activity, 3 out of 193 persons carry out about 70% of the total number of commits. Even more striking is the fact that a single developer accounts for 60% of the total number of commits. For the mail activity, 7 out of 92 persons sent about 60% of all the mails. For the bug report change activity, 5 out of 253 persons carry out about 40% of all bug report changes.

While a detailed statistical analysis of the exact type of distribution is left for future work, we do find clear support for the Pareto principle: for the commit activity, 20% of the most active committers contribute to about 85% of all commits. For the mail (resp. bug report change) activity, 20% of the most active committers contribute with about 75% of all mails (resp. bug report changes).

Figure 2 compares the activity distribution between different types of bug report activities available in the bug tracker: submitting new bug reports, changing existing bug reports, and commenting on existing bug reports. It provides similar

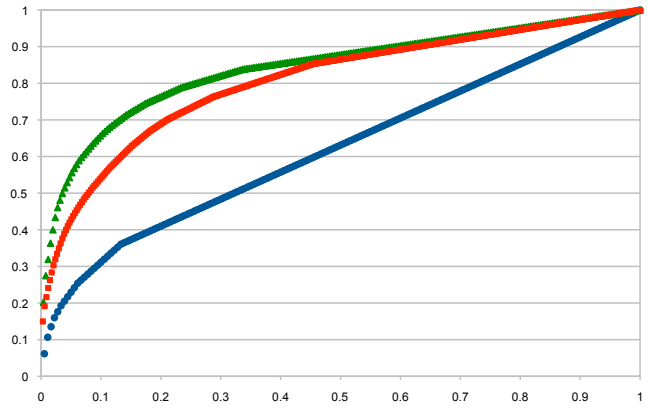

Fig. 2. Cumulative view of distribution of activity for Brasero bug reports (November 2010). The x-axis shows the cumulative percentage of active bug reporters, ordered from most to least active; the y-axis shows the cumulative percentage of bug report activity. The upper distribution (green triangles) corresponds to bug report changes, the middle one (red squares) to bug report submissions, and the lower one (blue circles) to bug report comments.

distributions as those found in Figure 1.

To answer research question 3 of Section II-A, we determined the overlap between different categories of activities (committing, mailing, changing bug reports), by analysing those persons that were involved in different activities. Figure 3 presents the results of this analysis. The triplet notation $(c\%, m\%, b\%)$ used in each of the intersections corresponds to the percentage of activity of a particular individual that contributed to more than one activity category. For example, there is a person with $(61\%, 11\%, 20\%)$, indicating that he contributed to 61% of the total activity of the top 20 most active committers, to 11% of the mail activity of the top 20 most active mailers, and to 20% of the bug report change activity of the top 20 most active bug report changers.

As expected, we find a clear overlap of activity. The two most active committers (out of the top 20) are also very active mailers and bug report changers. In fact, the same two persons account for 67% of the top 20 commit activity, 34% of the top 20 mail activity and 27% of the top 20 bug report change activity in Brasero. We also observe that three of the 20 most active mailers are also active as top 20 bug report changers.

Note that the analysis process for obtaining the results in Figure 3 was manual, which explains the restriction to the top 20 most active individuals only. The reason is that it is quite challenging to automate a reliable identification of identities (logins, e-mail addresses, names) that correspond to the same person. As can be seen in Figure 3, the total sum of all contributors per activity category is 19 instead of 20 for committers and mailers. The reason for this is that, during the manual analysis, we observed that two different identities actually corresponded to the same individual (because he has used two different e-mail addresses or logins over time). Therefore, we merged the corresponding data into a single entity.

To find out how the distribution of activity evolves over

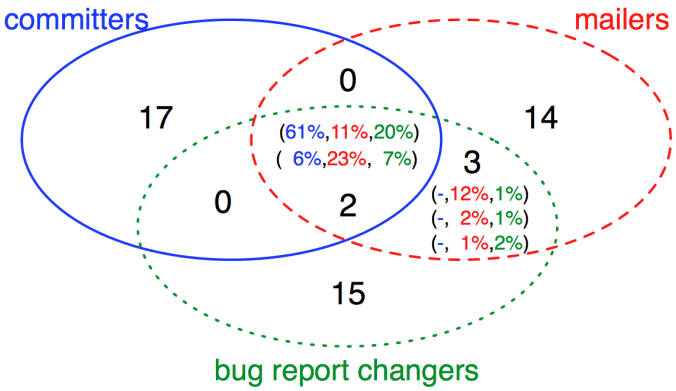

Fig. 3. Overlaps of activity for the top 20 most active individuals of the Brasero community for the three considered categories of activity (November 2010).

time, we used the econometric aggregation measures introduced in Section II-C. Figure 4 displays the evolution of three indices (Hoover, Gini and Theil) for the commits in Brasero. Each data point in this figure corresponds to a different distribution such as the ones shown in Figure 1. We observe that, regardless of the index used, the values do not fluctuate a lot, and tend to stabilise over time. For Gini, for example, we see that the index remains most of the time between 0.8 and 0.9, indicating a very unequal distribution of commit activity for all observed versions. This corroborates what we already observed before: a low number of individuals contribute most of the commits.

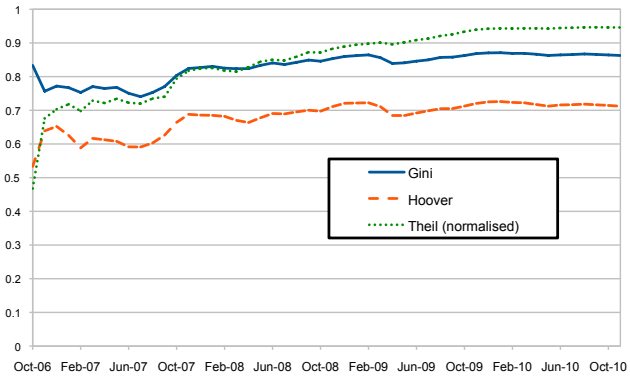

Fig. 4. Comparison of three aggregation indices, Gini (blue straight line), Theil (green dotted line) and Hoover (red dashed line), applied to the evolution of commit activity for Brasero since October 2006.

Figure 5 shows how the Gini index differs across the different activity categories we analysed over time for Brasero: commits, mails and bug report changes. Again, the results correspond to what we observed in the distributions of Figure 1. In all cases, the activity is unequally distributed across individuals. This is especially the case for the commits (with a single committer accounting for 60% of the total number of commits), explaining the high value of the Gini index. For

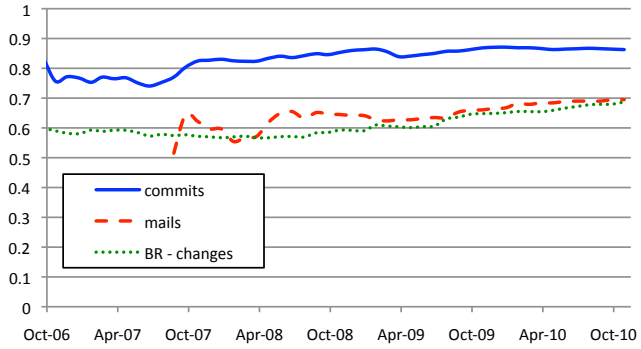

Fig. 5. Comparison of Gini indices over the evolution of Brasero commits (blue straight line) and bug report changes (green dotted line) since October 2006, and mails (red dashed line) since September 2007.

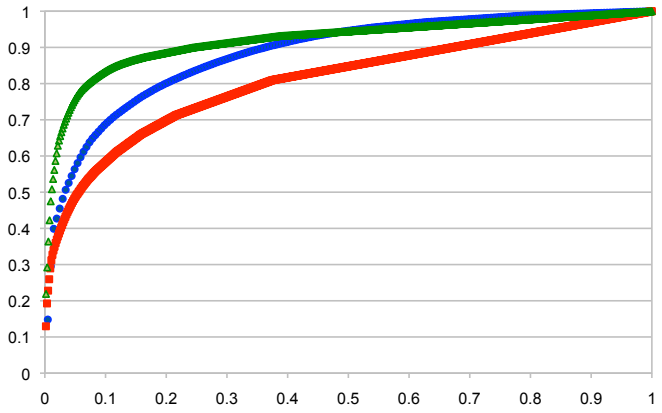

Fig. 6. Cumulative distribution of activity for Evince (November 2010). The distribution with blue circles corresponds to the commit activity. The distribution with red squares corresponds to the mail activity. The distribution with green triangles corresponds to the bug report change activity.

mail activity and bug report change activity, there is also an unequal distribution, but less flagrant than for the commits. This explains why their Gini index curve is below the one for the commit activity.

B. Evince

A study of the evolution of the Evince community provides similar results. Perhaps an important difference is that the development activity is a bit more equally distributed than for Brasero (where we found a single person responsible for 60% of the total commit activity).

Figure 6 shows the cumulative activity distribution for three types of activity for Evince: commits to the version repository, mails sent to the mailing list, and changes made to bug reports in the bug tracker. Evince has two top committers, each accounting for 15% of the total commit activity. The Pareto principle is also clearly present in Figure 6: 20% of all committers contribute to 80% of the total commit activity, 20% of all mailers contribute to 70% of the total mail activity, and 20% of all bug report changers contribute to 88% of the total bug report change activity.

Figure 7 provides a different view on the same data, obtained by manually analysing the top 20 of most active persons for each activity category. As for Brasero, the 4 most active persons of the Evince community contribute to each of the three activity categories. Together, they account for 35% of all top 20 commits, 26% of all top 20 mails, and 36% of all top 20 bug report changes.

Figure 8 shows the evolution of the three econometric aggregation indices on the evolution of Evince's commit activity. After a steep startup phase, we see that the indices start to stabilise rapidly to a more or less stable value. This value is lower than for Brasero, since the distribution of commit activity is a bit more equally distributed for Evince.

Since Figure 8 reveals a similar pattern for all three indices, in Figure 9 we restricted ourselves to the Gini index only. We used it to compare the evolution of the commit, mail and bug report activity of Evince. For each of these activities, we observe that the Gini index stabilises rapidly to a more or less

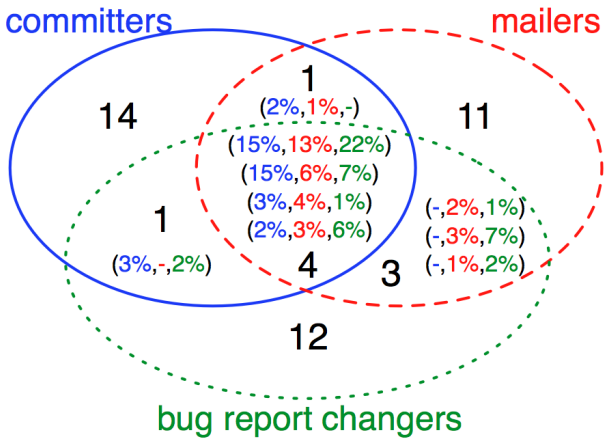

Fig. 7. Overlaps of activity for the top 20 individuals contributing to the considered Evince activity categories (November 2010).

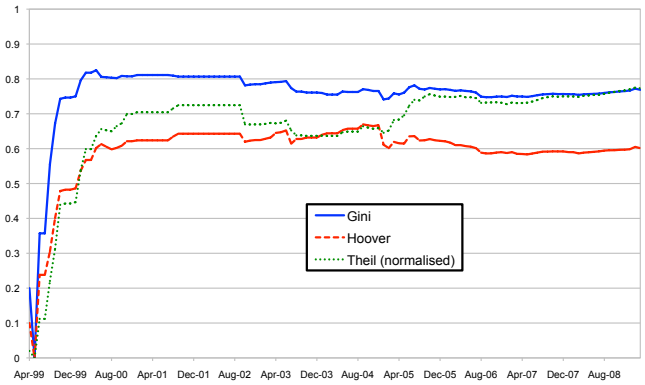

Fig. 8. Comparison of three aggregation indices applied to the evolution of commit activity for Evince since April 1999.

constant value, indicating that the way in which the community is structured is fairly stable. For commits and mails, the high Gini index indicates that the activity is not equally distributed over the community members. For mails, the Gini index is lower so this activity is more spread over different persons.

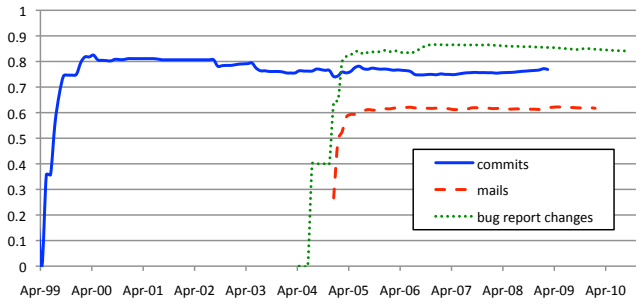

Fig. 9. Comparison of the evolution of Gini indices for Evince, since April 1999 for commits (blue straight line), since January 2005 for mails (red dashed line), and since August 2004 for bug report changes (green dotted line).

C. Wine

A study of the Wine community reveals that the number of persons involved is much bigger than for the other two systems studied. This was already apparent from Table I.

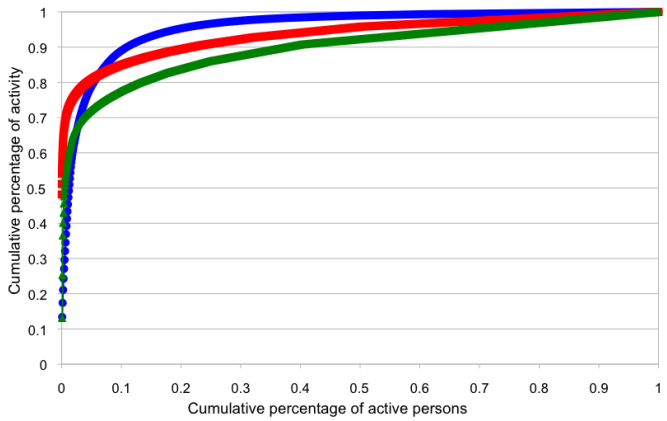

Fig. 10. Cumulative distribution of activity for Wine (November 2010). The distribution with blue circles corresponds to the commit activity. The distribution with red squares corresponds to the mail activity. The distribution with green triangles corresponds to the bug report change activity.

Figure 10 shows the cumulative activity distribution for Wine committers, mailers and bug report changers. The most active committer accounts for 13% of the total commit activity, the two most active bug report changers each account for 13% and 11% of the total change activity, respectively.

Concerning the mail activity of Wine, we observed that there is one huge mailer `wineforum-user@winehq.org` accounting for 48% of the total project mail activity. It turned out that this mailer was in fact an automated transcription of the discussion forum in the mail system. After excluding this outlier from the mail activity data set, we found that the most active mailer only accounts for 4% of the total mail activity.

Figure 10 also provides evidence for the Pareto principle. 11% of all committers account for a total of 90% of commits. Similarly, a total of 13% of bug report changers account for 80% of all changes.

As a side note, we observed that the use of logins and accounts in Wine was poorly structured. For example, many committers have 4 or 5 email addresses that are rarely used. This may be explained by the fact that the Wine community is very open: it is very easy for new persons to become active in this community.

Figure 11 shows the overlap of activity of the top 20 most active persons in each activity category for Wine. Again, we needed to merge two different identities corresponding to the same individual into a single identity (explaining a total sum of 19 instead of 20 for committers). In contrast to the previously analysed projects, none of the 20 most active persons contributed to three different activity categories. Another major difference was that the core group of active persons for Wine was significantly bigger, explaining the smaller percentages we obtained for the most active persons involved in a particular activity.

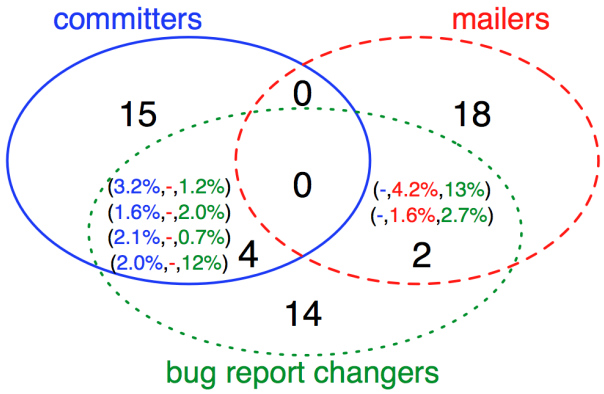

Fig. 11. Overlaps of activity of top 20 individuals contributing to the considered Wine activity categories (November 2010).

Figure 12 compares the evolution of Wine's commit activity using three different aggregation indices. We observe that the Gini and Theil indices are very high and continue to increase over time, indicating a very unequal distribution of activity, with a large group of inactive persons and a small group of active persons.

Figure 13 displays the evolution of the three types of activity for Wine over time, using the Gini index as aggregation measure. For mail activity, the Gini index is initially much lower than for the commit activity, but after a while its value starts to increase to comparable values. This high value is largely explained by the presence of a single artificial mailer (corresponding to the wineforum). For the bug report change activity, we also observe an increasing growth of the Gini index, revealing an increasing inequality of activity distribution over time.

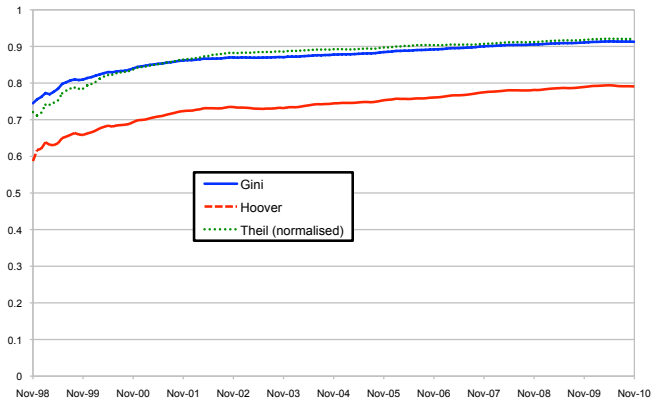

Fig. 12. Comparison of three aggregation indices applied to the evolution of commit activity for Wine since November 1998.

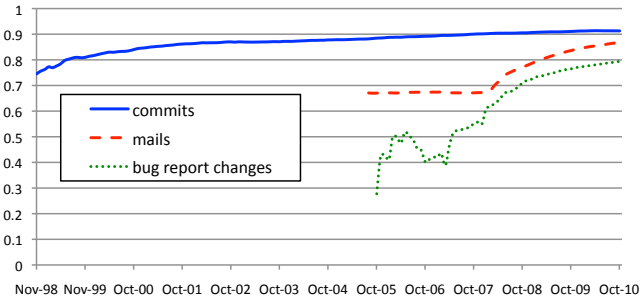

Fig. 13. Comparison of the evolution of Gini indices for Wine, since November 1998 for commits (blue straight line), since September 2005 for mails (red dashed line), and since November 2005 for bug report changes (green dotted line).

V. Discussion

The results shown in the previous section provide strong evidence for the Pareto principle. The activity of contributors to open source projects is not equally distributed: in all three studied projects, a core group of persons appears to carry out the majority of the work. This kind of behaviour may be related to the way in which open source developer communities are structured. According to [12], a typical structure is the so-called onion model. It is followed, among others, by the community in charge of the Linux kernel. In such a layered model, there is a single responsible of the project, surrounded by a small core team of software developers, around which there is a bigger layer of active developers, followed an even bigger layer of occasional developers. The last layer constitutes those users of the project that do not contribute anything themselves. The more to the center of the onion, the more active a developer, and conversely. Although many variants of this layered model exist, the general idea behind it remains the same.

Our empirical analysis showed the usefulness of applying results from econometry to the analysis of the activity in software ecosystems. We used three different aggregation indices of statistical dispersion, the Hoover index, Theil index and Gini index. They all gave similar results, i.e., they tend to evolve in the same way for a given OSS project's history. This probably implies that one can freely choose any of these indices to assess the evolution of activity distribution. This corresponds to the findings of Vasilescu et al. [13] that compared different aggregation measures applied to software metrics and observed a strong correlation between them.

For all three OSS projects we studied, the distribution of activity was initially more equally distributed, but over time the activity tends to become concentrated in a core group of persons that is significantly more active than the others. This knowledge is quite important, as the sudden disappearance of some members of the core group may have an important impact on the future of the software project. In other ways, we found empirical evidence of the so-called *bus factor*, the total number of key persons that would, if they were to be hit by a bus, lead the project into serious problems. Note that this was less the case for Wine, by far the biggest of the three projects, where the activity was more equally distributed over the most active committers than for the other two projects.

For all types of activities, we found a long tail of persons whose activity rate can be largely neglected: during the entire lifetime of the project, they contributed once or twice to one of the considered project activity categories (commits, mails and bug reporting). We also observed that, except for Wine, the most active project members take part in all these activities.

As a potential threat to validity, during our experiments, we encountered some problems to determine which persons contribute to different activity categories (Figures 3, 7, 11). This was mainly a manual and error-prone process that took a lot of effort. In the future we intend to automate this process. To achieve this, an efficient identity merging algorithm is needed that allows one to identify matches between entities participating in different data sources. But for poorly structured projects such as Wine this may still be quite problematic and difficult to automate. We have studied this topic in more detail in [14].

VI. Related Work

Some researchers focus on the study of core teams [15], [16] and the inequality of distribution in software development [13], [3], [4]. Our work distinguishes itself from that of most other researchers involved in mining software repositories [17], [18], [19], [20], [21], who tend to focus on the analysis of the software development *artefacts* (e.g. source code, bug reports, and so on) and the dependencies between those. Instead, our main interest goes to the *individuals* involved in creating and modifying those artefacts, as well as the interaction and communication between those individuals.

Because the data related to software development activity is dispersed and incomplete, and due to the heterogeneity of data formats used in software development, an important effort must be made in order to collect and represent all available data in a uniform way [22], [23]. In previous work [14], we discuss the need to merge identities (logins, e-mail addresses,

names) across different data sources, and compare existing identity merging algorithms that try to achieve this.

Numerous studies have found evidence for the Pareto principle (or, more generally, a power law distribution) in human-related networks [24]. For instance, evidence for a power law distribution has been found in the number of citations in papers [25], the number of sexual partners in human societies [26], and many more. In software evolution, Herraiz founds a double Pareto distribution in software size [8] (using different measures of size). Mitzenmacher generalized this observation for file system distributions [27]. Hunt and Johnson demonstrated [28] that most of the data available in Sourceforge, a software forge for free/open source software[2], follows a Pareto distribution.

VII. Conclusion

In this article, we studied and compared the evolution of OSS project activity. Following the GQM paradigm, our main research goal was to understand how activity is distributed in OSS projects over time. We considered three categories of activity: committing data and code, sending mail and changing bug reports. We extracted and analysed such activity based on three different types of data sources: version repositories, mailing lists, and bug tracking data. We carried out an empirical analysis over three different long-lived OSS projects for which this data was available: Brasero, Evince and Wine.

For all three studied projects and for all considered activity categories, we found evidence for the Pareto principle. The activity distributions showed a strong inequality in the activity of different persons involved in an OSS project: there is a small group of very active members, and a much bigger group of largely inactive members. For two of the three studied projects, the core group of most active members takes part in more than one activity category. In Wine this was much less the case.

In order to gain understanding in how OSS projects evolve, we studied this inequality of distribution over time. To do so we relied on statistical techniques borrowed from econometrics. We applied three economic aggregation measures (the Hoover, Gini and Theil index). The evolution of activity distribution appeared to follow two kinds of behaviour. The first one is typical of a totally new project: at the beginning, the activity is more or less equally distributed, but quickly we observe a tendency towards a more unequal distribution where the activities become more concentrated in a small core team. In the second type of observed behaviour, the activity distribution is already imbalanced since the beginning of the project, and this imbalance continues to become more pronounced over time.

Studying who are the most active persons involved in each type of activity, we discovered that these persons are often very active in different activity categories. For Brasero and Evince, the two projects in which we observed this behaviour, the project is led by a small group of very active members wearing several hats at the same time. We have not yet been able to identify the cause of overlaps between activities, because our definitions for measuring activity need to be refined further.

While Brasero and Evince show a similar evolution of activity distribution and similar overlaps between most active persons' activity categories, the Wine software project appears to have a different behaviour. It has a significantly bigger community, there is significantly less overlap between activity categories, and we observed a higher inequality in the activity distribution. We can only speculate as to the causes of this.

VIII. Future work

While the work presented in this article is very promising, many challenges lay still ahead of us.

We intend to carry out a detailed statistical study of the activity distributions we presented in this article. In particular, we would like to find out which kind of statistical distribution they represent. Can we find evidence for power laws, Pareto distributions or other types of statistical distributions?

We also intend to analyse the activity distribution for subgroups and subprojects, and study how the structure and size of the project community and the software itself affects the type of activity distribution. Another aspect worthy of further study is the evolution of the activity of individuals involved in OSS projects. Can we find evidence for a learning curve for newcomers in a project? Does the activity of core members increase or decrease over time? Can we identify certain typical evolution patterns of activity?

Within an OSS project we would like to study the statistical correlation between different characteristics such as software quality, stakeholder activity, and software size. We also want to study how each of these characteristics correlate across different projects.

The notions of activity studied in this article, and the metrics used for computing them, are perhaps too high-level. We could define and use more fine-grained and more specific definitions, in order to get a more detailed picture of how project members interact, and in order to come up with new effort estimation and effort prediction models based on the activity of project members.

It remains an open question whether the type of activity distribution we observed in our empirical study is specific to open source projects. Our hypothesis is that this is indeed the case. To verify this hypothesis, we would like to repeat our experiment on proprietary, closed source software projects. The main challenge here it to get access to such data, in particular, the evolutionary data related to the activities of the developers involved in these projects.

Finally, the long term goal is to provide assistance or guidance, through interactive or automated tool support, for all stakeholders involved in an (open source) software project. Users may rely on information such as the bus factor to decide on using a particular open source software product. Developers may use it to identify the core developers and influential persons. Project managers may wish to control the activity distribution, to estimate or predict the effort, or to reduce the bus factor risk through a better distribution of activity.

[2]http://sourceforge.net/

ACKNOWLEDGMENT

The research is partially supported by (i) F.R.S.-FNRS FRFC project 2.4515.09 "Research Center on Software Adaptability"; (ii) research project AUWB- 08/12-UMH "Model-Driven Software Evolution", an Action de Recherche Concertée financed by the Ministère de la Communauté française - Direction générale de l'Enseignement non obligatoire et de la Recherche scientifique, Belgium.

REFERENCES

[1] J. Fernandez-Ramil, A. Lozano, M. Wermelinger, and A. Capiluppi, "Empirical studies of open source evolution," in *Software Evolution*, T. Mens and S. Demeyer, Eds. Springer, 2008, pp. 263–288.

[2] V. R. Basili, "Software modeling and measurement: the goal/question/metric paradigm," College Park, MD, USA, Tech. Rep., 1992.

[3] R. Vasa, M. Lumpe, P. Branch, and O. Nierstrasz, "Comparative analysis of evolving software systems using the Gini coefficient," in *Proc. Int'l Conf. Software Maintenance*, 2009, pp. 179–188.

[4] A. Serebrenik and M. van den Brand, "Theil index for aggregation of software metrics values," in *IEEE International Conference on Software Maintenance*. Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 1–9.

[5] H. Theil, *Economics and information theory*. Center Math. Stud. Business Econ., Univ. Chicago, 1967.

[6] M. Goeminne and T. Mens, "A framework for analysing and visualising open source software ecosystems," in *Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE)*, ser. IWPSE-EVOL '10. New York, NY, USA: ACM, 2010, pp. 42–47.

[7] M. Newman, "Power laws, Pareto distributions and Zipf's law," *Contemporary Physics*, vol. 46, no. 5, pp. 323–351, 2005.

[8] I. Herraiz, "A statistical examination of the evolution and properties of libre software," Ph.D. dissertation, Universidad Rey Juan Carlos, 2008.

[9] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan, "Mining email social networks," in *MSR '06: Proceedings of the 3rd IEEE International Working Conference on Mining Software Repositories*, A. Press, Ed., Shanghai, China, May 2006.

[10] M. Hardy, "Pareto's law," *The Mathematical Intelligencer*, vol. 32, pp. 38–43, 2010, 10.1007/s00283-010-9159-2.

[11] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009.

[12] M. Antikainen, T. Aaltonen, and J. Vaisanen, "The role of trust in OSS communities - case Linux Kernel community," in *Open Source Development, Adoption and Innovation*. Springer, Jun. 2007, pp. 223–228.

[13] B. Vasilescu, A. Serebrenik, and M. van den Brand, "Comparative study of software metrics aggregation techniques," in *BENEVOL 2010*, December 2010.

[14] M. Goeminne and T. Mens, "A comparison of identity merging algortihms for open source software ecosystems," *Journal on Systems and Software. [Submitted]*, 2011.

[15] G. Robles, J. M. Gonzalez-Barahona, and I. Herraiz, "Evolution of the core team of developers in libre software projects," in *MSR '09: Proceedings of the 6th IEEE International Working Conference on Mining Software Repositories*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 167–170.

[16] S. Minto and G. C. Murphy, "Recommending emergent teams," in *Proceedings of the Fourth International Workshop on Mining Software Repositories*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 5–.

[17] M. D'Ambros, H. Gall, M. Lanza, and M. Pinzger, "Analysing software repositories to understand software evolution," in *Software Evolution*, T. Mens and S. Demeyer, Eds. Springer, 2008, pp. 37–67.

[18] S. Diehl, H. C. Gall, and A. E. Hassan, Eds., *Special Issue on Mining Software Repositories*, ser. Empirical Software Engineering, vol. 14, no. 3, Jun. 2010.

[19] R. Abreu and R. Premraj, "How developer communication frequency relates to bug introducing changes," in *IWPSE-Evol '09: Proceedings of the joint international and annual ERCIM workshops on Principles of software evolution (IWPSE) and software evolution (Evol) workshops*. New York, NY, USA: ACM, 2009, pp. 153–158.

[20] D. M. German, "Mining cvs repositories, the softchange experience," in *Proceedings of the First International Workshop on Mining Software Repositories*, Edinburg, Scotland, UK, 2004, pp. 17–21.

[21] W. Poncin, A. Serebrenik, and M. van den Brand, "Process mining software repositories," in *CSMR '11: Proceedings of the European Conference on Software Maintenance and Reengineering.*, 2011.

[22] I. Herraiz, G. Robles, and J. M. Gonzalez-Barahona, "Research friendly software repositories," in *IWPSE-Evol '09: Proceedings of the joint international and annual ERCIM workshops on Principles of software evolution (IWPSE) and software evolution (Evol) workshops*. New York, NY, USA: ACM, 2009, pp. 19–24.

[23] I. Herraiz, D. Izquierdo-Cortazar, and F. Rivas-Hernández, "Flossmetrics: Free/libre/open source software metrics," in *CSMR '09: Proceedings of the 2009 European Conference on Software Maintenance and Reengineering*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 281–284.

[24] M. E. J. Newman, "The Structure and Function of Complex Networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[25] S. Redner, "How popular is your paper? An empirical study of the citation distribution," *The European Physical Journal B*, vol. 4, p. 131, 1998.

[26] F. Lilijeros, C. Edling, L. Amaral, E. Stanley, and Y. åberg, "The web of human sexual contacts," *Nature*, vol. 411, pp. 907–908, 2001.

[27] M. Mitzenmacher, "Dynamic models for file sizes and double Pareto distributions," *Internet Mathematics*, vol. 1, pp. 305–333, 2002.

[28] F. Hunt and P. Johnson, "On the Pareto distribution of Open Source projects," in *Proceedings of Open Source Software Development Workshop*, Newcastle, UK, 2002.