Evolution of Volunteer Participation in Libre Software Projects: Evidence from Debian

Gregorio Robles, Jesus M. Gonzalez-Barahona
Grupo de Sistemas y Comunicaciones
Universidad Rey Juan Carlos (Madrid, Spain)
{grex,jgb}@gsyc.escet.urjc.es

Martin Michlmayr
Centre for Technology Management
University of Cambridge
martin@michlmayr.org

Abstract— **Most libre software projects rely on the work of volunteers. Therefore, attracting people who contribute their time and technical skills is of paramount importance, both in technical and economic terms. This reliance on volunteers leads to some fundamental management challenges: volunteer contributions are inherently difficult to predict, plan and manage, especially in the case of large projects. In this paper we analyze the evolution in time of the human resources of one of the largest and most complex libre software projects composed primarily of volunteers, the Debian project. Debian currently has around 1300 volunteers working on several tasks: much activity is focused on packaging software applications and libraries, but there is also major work related to the maintenance of the infrastructure needed to sustain the development. We have performed a quantitative investigation of data from almost seven years, studying how volunteer involvement has affected the software released by the project and the developer community itself.**

Index Terms— **libre software engineering, human resources, volunteer developers, software evolution.**

I. Introduction

Volunteer contributions are the base of most libre software projects[1]. However, the characteristics and way of working of volunteers can be quite different from those of employees, which are the main force behind traditional software development. Volunteers can contribute with the amount of effort they want, can commit for the time period they consider convenient, and can devote their time to the tasks they may prefer, given that the context of the project allows them to do so. Despite of this, some libre software projects have produced software which has gained significant popularity. This shows that the unstructured collaboration of volunteers is a viable software development strategy, even if it is associated with certain challenges related to project management and quality. In this paper we will explore how these voluntary contributions have been working in the specific case of a large libre software project, to have some actual data about their behavior. We started the study stating some questions, for some of which we thought we already knew the answer. To our surprise, we found out that, even with the high knowledge we thought we had about the history of the project, the data told a different tale.

We define in this paper volunteers as those who collaborate in libre software projects in their free time not profiting economically in a direct way from their effort. Volunteers can be IT-related professionals or not, but their professional activity is not the one they perform on a given libre software project. Although the vast majority of participants in libre software projects are following our definition volunteers there also exist non-volunteers (also known as paid employees), i.e. those whose professional activity is to work on that specific project. In a study on the GNOME project [1], German states that paid employees from various companies are usually responsible for less attractive tasks, such as project design and coordination, testing, documentation and bug fixing. Also, "[m]ost of the paid developers in GNOME were, at some point, volunteers. Essentially for the volunteers, their hobby became their job" [1].

The involvement of volunteers, of course, raises new economic issues that have to be taken into account for business strategies around libre software. Collaboration from volunteers is difficult to predict, but if it is given it may add value to a software system in very economic terms for a software company.

The structure of this paper is as follows. The next section briefly explains the nature of maintainers in the Debian project, and of the Debian operating system. Then we state the questions we aimed to answer before starting this study, including possible answers we expected at that time. After that, we explain the methodology we devised and followed to answer those questions, and the data sources we used. We later show the actual results obtained from the empirical analysis, and contrast them with what we had expected. The paper ends with some conclusions and lessons learned.

II. Maintainers in the Debian project

Debian is an operating system completely based on libre software [2], [3]. It includes a large number of applications, such as the GNU tools and Mozilla, and the system is known for its solid integration of different software components. Debian's most popular distribution, Debian GNU/Linux, is based on the Linux kernel. Ports to other kernels, such as the Hurd and FreeBSD, are in development.

One of the main characteristics of the Debian distribution is that during the whole life of the project it has been maintained by a group of volunteers, which has grown to quite a substantial number. These individuals devote their own time and technical skills to the creation and integration of software packages, trying to supply users with a robust system which provides a lot of functionality and technical features.

One of the main characteristics of the Debian distribution is that the bulk of work has always been performed by volunteers; furthermore, the project has grown to substantial

[1] In this paper we will use the term "libre software" to refer to any software licensed under terms compliant with the FSF definition of "free software", and the OSI definition of "open source software", thus avoiding the controversy between those two terms. However, in the specific case of Debian, the project has its own definition of "free software", the Debian Free Software Guidelines, from which the OSI definition originated later.

Proceedings of the First International Conference on Open Source Systems
Genova, 11th-15th July 2005
Marco Scotto and Giancarlo Succi (Eds.), pp. 100-107

size over the years. The individuals involved in Debian devote their own time and technical skills to the creation and integration of software packages, thereby supplying users with a robust system which provides a lot of functionality and technical features.

Following our definition of volunteers, all maintainers in Debian are volunteers. Some employers of people who act as Debian maintainers in their spare time permit their staff to devote some of their time to Debian during work hours. Nevertheless, the majority of work by most Debian maintainers is performed in their spare time. In contrast to some projects, such as the Linux kernel and GNOME, there are no Debian maintainers who are paid to work on the system full-time.

There are several tasks that volunteers can do in Debian: maintaining software packages, supporting the server infrastructure, developing Debian-specific software as for instance the installation and package management tool, translation of documentation and web pages, etc. From all these tasks, we will focus in this paper on maintainers, whose task is to take existing libre software packages and to create a ready-to-install Debian package. Debian maintainers are also called Debian developers, although their task is really not to develop software but to take already developed software for creating a package. This, of course, does not mean that a Debian maintainer may not develop and maintain a software, but this is not usually the case: the original author (or developer) and the Debian maintainer are commonly not the same persons.

Besides its voluntary nature, the Debian project is unique among libre software projects because of its Social Contract [4]. This document contains not only the primary goals of the Debian project, but also makes several promises to its users. Additionally, there are a number of documents Debian maintainers have to follow in order to assure quality, stability and security of the resulting distribution. In particular, Debian's Policy document ensures that the large number of volunteers working independently will produce a well integrated system rather than merely an aggregation of software packages which do not play together very well [5].

The most common task performed by Debian maintainers is to maintain software packages. It involves obtaining the source code from its original authors (known as 'upstream'), compiling it and creating a Debian package by following certain rules in order to make the system consistent. It is unusual that the original software is written by a Debian maintainer, although some packages, such as Debian's package maintenance system `dpkg`, are. Debian maintainers generally take source code from others.

There has recently been some interest in studying how the voluntary nature of the Debian members affect the quality of the resulting product. Managing volunteer contributors is associated with certain problems that 'traditional' software development usually do not confront [6]. Since Debian maintainers are volunteers who perform their tasks in a distributed environment, some new aspects regarding the management of contributors have to be taken into consideration, such as the unpredictability of the level of their involvement [7]. To some degree, the volatility of voluntary contributors can be limited by the introduction of more redundancy, such as the creation of maintainer teams.

The creation of teams and committees for specific purposes such as management or for complex tasks has been already reported by German's work on the GNOME project [1].

III. Research targets

Given the importance and volume of the Debian project, it is worth studying the evolution of the people maintaining packages for it. This paper aims to give some quantitative insight about the evolution of Debian maintainers during a six and a half year period (from July 1998 to December 2004). We study how many of these maintainers have remained from the beginning, as well as what has happened to packages from manintainers that left the project. As stated before, of the many tasks performed in the Debian project (ranging from the administration of the infrastructure to the translation of help texts and documentation into a number of supported languages), only packaging activities have been taken into account. These activities are the more notorious, and most maintainers focus on them. They are also the most easy to quantify.

The specific questions that we wanted to answer with this study are the following:

A) How many maintainers has Debian, and how does this number change over time?
This will provide us with some basic data useful when working with subsequent questions. When we started the study, we expected a steady increase of maintainers over time, as it is already known that the number of packages included in the system has been growing that way [8]. In fact, we expected the packages to maintainers ratio to be somewhat constant, since it seems reasonable to consider that volunteers devote similar amounts of effort over time, which would lead to a constant number of packages per maintainer.

B) Is there a trend towards the formation of maintainer teams?
Building teams of maintainers for a package is one of the ways that has been proposed in Debian for lowering the problems of unpredictability of the maintainers [7]. The answer to this question should show whether this proposal has indeed been implemented. It could also have an impact on the ratio explored in the previous question. If maintainers devote more time to team-maintained packages (because there are more of them), that would mean they have less time for single-maintainer packages, and therefore have less of those. Which could lead to a decrease, in the latest release, of the packages to maintainers ratio.

C) How many maintainers from previous releases remain active?
We want to measure the volatility of the volunteers in the Debian project. That is, do maintainers join the project and work on it for short periods of time, or on the contrary do they stay for many years? Specifically, we have calculated the half-life of contributors in the project (half-life being the time required for a certain population of maintainers to fall to half of its initial size). This figure could be easily compared

with other libre software projects and, of course, with statistics from companies from the software and other industries.

Our study has been made in a per-release basis instead of investigating the evolution on a time basis, because we only have data available for each release. This conforms to the 'traditional' software evolution methodology by Lehman [9].

D) Which is the contribution of maintainers who remain in subsequent releases?

Answering this question will allow us to know if 'older' maintainers strengthen their contributions as time passes by, maintaining more packages, or whether they become less active. There are two possible hypotheses one could propose. On one hand, those maintainers who have been involved for a long time may be very experienced and therefore more efficient in their work than less experienced developers. On the other hand, young developers may have more time or energy to devote to the project and therefore contribute more. Both theories are possible and mutually compatible.

E) What happens to packages maintained by maintainers who leave the project?

Since they are volunteers, maintainers may leave the project almost anytime, leaving their packages unmaintained. There are two chances for those packages: being taken over (adopted) by other maintainers, or being excluded from future stable releases. As those may not fulfill the requirements of the Debian policy and the quality standards of the project, they may be removed from the next release if they are not adopted by other maintainers. Our intention was to know how this inherent characteristic of the voluntary contributors affect Debian, and how this is damped down by other (possibly new) maintainers. In a sense this question targets how well Debian regenerates itself and survives the loss of some of its human resources.

F) Are more 'important' and commonly used packages maintained by more experienced maintainers?

It can be interesting to know whether packages which are considered crucial for the functioning of the system are maintained primarily by volunteers who have more experience with Debian. For this, we considered the most used packages as the targets of the study. It seemed reasonable to think that the answer would be "yes", since probably most used packages were introduced in earlier releases. In any case, this could lead to have some insight on whether there is some specialization of veteran maintainers in packages with more impact on the user experience of the system, which because of our experience with the project seemed a reasonable assumption.

Crucial packages are those packages usually installed on every system, as for instance the base system which is composed among others of the Linux kernel and the GNU tools for the Debian GNU/Linux operating system. This does not mean, of course, that crucial packages are more difficult to maintain than other packages, but as they are used by any user of the system and the rest of the software heavily depends on its proper functioning, these packages have to be maintained with special care.

IV. Methodology and sources of data

Debian consists of four parallel versions (stable, testing, unstable, experimental) which can be downloaded from the Internet. The focus of this study is on the stable versions from Debian 2.0 onwards up to 3.0, and on the current candidate for Debian 3.1 (as of December 4, 2004)[2], which provide good snapshots of the history of the distribution. This means a period of time from July 1998 to December 2004. There have been releases of Debian before 1998 [10], but they have not been taken into consideration for this study, since the sources of data we have used were not available for them.

For each release, we have retrieved the corresponding Sources.gz file (see below) from the Debian archive. From it, we have extracted the information about the packages and their maintainers, which we stored in a database. After that, we performed some semi-automatic cleaning and massaging of the data that will be explained in more detail below. Final results were obtained through queries to the database, and correlations that have been implemented by another set of scripts[3].

The estimations of the size of the releases have been done using the methodology described in [11], using data already published in [8], except for release 3.1, which was calculated specifically for this study.

The data related to the importance of packages has been retrieved from the Debian Popularity Contest (see below).

A. The Sources.gz file

Since 2.0, the Debian repository contains a Sources.gz file for each release, listing information about every source package in it. For each package, it contains: name and version, list of binary packages built from it, name and e-mail address of the maintainer, and some other information which is not relevant for this study.

For identifying the contribution of individual volunteers we had to find out when a package is maintained not by a single person, but by a team, and consider them separately. For that, we merged entries for packages maintained by a groups to a unique one called 'Debian team'. We later use the data for that entry only for answering question 2, while for the rest of the results we will pay attention just to individual contributors.

As an example, see below an excerpt of the entry of the source package for Mozilla from Debian 2.2. It can be seen among other information how this package corresponds to version M18-3, provides four binary packages (for the sake of space the last one is not shown), and is maintained by Frank Belew.

[2] At the time of writing, the next stable release, 3.1, is still in preparation. This is why '3.1' will appear in parenthesis in some tables. Since the release of 3.1 is considered to be imminent, we believe that the data included in this study for the current candidate will be quite similar to those of the final version.

[3] All the code used has been released as libre software, and can be obtained from `http://libresoft.dat.escet.urjc.es/index.php?menu=Tools`

```
[...]
Package: mozilla
Binary: mozilla,mozilla-dev,libnspr4,...
Version: M18-3
Priority: optional
section: web
Maintainer: Frank Belew <frb@debian.org>
Architecture: any
Standards-Version: 3.2.0
Format: 1.0
Directory: dists/potato/main/source/web
[...]
```

B. *Debian Popularity Contest*

The Debian Popularity Contest is an attempt to map the usage of Debian packages. Its main goal is to know what software packages are actually installed and used.

This system functions as follows: Debian users may install the `popcon` package which sends a message every week with the list of packages installed on the machine as well as the access time of some files which may give a hint of the last usage of these packages. Of course, privacy issues are considered in a number of ways: upon installation, the user is explicitly asked if she wants to send this information to Debian, and the server which collects the data anonymizes it as much as possible.

The resulting statistical information of all users participating in this scheme is publicly available on the web site of the project. For every package it includes the number of machines on which it is installed (inst), the number of machines which make regular use of that package (vote), the number of recent updates (recent), the number of machines where not enough information is available (no-file) and the maintainer of the package. Below is an excerpt of the available data, in this case the top ten packages ordered by number of installations as of December 4th, 2004.

rank	name	inst	vote	old	recent	no-files
1	adduser	6881	6471	94	316	0
2	debianutils	6881	6517	50	314	0
3	diff	6881	6425	261	195	0
4	e2fsprogs	6881	5448	825	608	0
5	findutils	6881	6449	233	199	0
6	grep	6881	6436	126	319	0
7	gzip	6881	6558	245	78	0
8	hostname	6881	6112	715	54	0
9	login	6881	6407	56	418	0
10	ncurses-base	6881	56	143	6	6676

TABLE I

EXCERPT FROM THE DEBIAN POPULARITY CONTEST. FIRST TEN PACKAGES BY INST(ALLATIONS). THERE IS ALSO A MAINTAINER COLUMN WHICH HAS BEEN OMITTED FROM THIS TABLE FOR THE SAKE OF SPACE.

The Debian Popularity Contest gives Debian a way to see which packages are in use. This information is used in order to determine the order in which packages are put on different CDs (i.e. packages with a high popularity and usage are put on the first or first few CDs); it is also used during quality assurance activities as a criteria on which packages to focus.

C. *Some constraints*

While the acquisition of data is straightforward because of its public nature, there are some tasks that had to be performed to ensure the correctness of our results. One of the main problems we faced was the identification and merge of the different entries for the same maintainer, due to changes in the e-mail address or spelling of the name (addition of the middle name, nickname, etc.).

We also had to resolve inconsistencies between the information in the Sources.gz file and the data from the Debian Popularity Contest. While the former contains entries for source packages, the latter tracks statistics for binary packages. Fortunately, we could link the source package to its binary packages by using the information from Sources.gz.

V. RESULTS

After applying the methodology described in the previous section, we are presenting in the next subsections the answers the questions that motivated this study.

To provide some context for these results, figure 1 shows the evolution of the size of Debian (both in lines of code and in packages) for the releases studied. In it the reader can find out how by both metrics the total size of the distribution seems to double approximately every two years.

A. *Number of maintainers*

Figure 2 shows the evolution of the number of Debian maintainers for the latest five stable releases. As we had expected, the number of Debian individual maintainers has been growing in time. 2.0 (July 1998) was put together by 216 individual maintainers, while the number of maintainers for later releases are 859 for 3.0 (July 2002) and will be of about 1,237 for 3.1 (December 2004). This shows a growth of about 35% every year.

However, the ratio of packages per maintainer grows over time, contrary to our initial hypothesis. The growth of packages is actually bigger than that of volunteers contributing. It is possible that this finding is related to improvements of development tools or practices which have increased the efficiency of package maintainers.

B. *Debian Maintainer Teams*

The data from table II shows a clear increase in the number of maintainer teams in Debian. The number of packages maintained by more than one maintainer has grown from 14 (1.3%) to almost 600 (7.4%) during the last six and a half years. This means that the awareness of the Debian project about the unpredictability of volunteers has had a response in the creation of these teams, especially since 2002.

However, it is important to highlight that a very special team, the QA (quality assurance) team "inherits" packages without maintainer, and therefore those are not really maintained packages (that is why we offer a column for those packages in table II). But even neglecting the contribution

Date	Rel	Size	Pkgs
Jul98	2.0	25	1,115
Mar99	2.1	37	1,580
Aug00	2.2	59	2,647
Jul02	3.0	105	5,220
Dec04	(3.1)	196	8,385

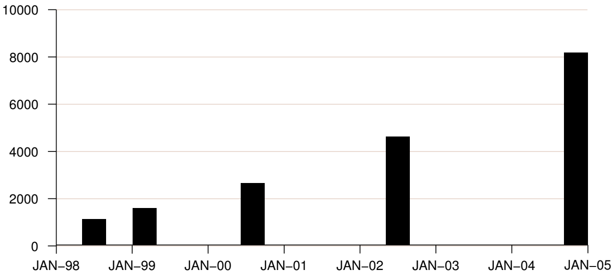

Fig. 1. Last five stable Debian releases. On the left, size (in millions of source lines of code) and total number of packages. On the right, number of packages in each release over time.

Date	Rel	Mnt	Pkgs	Pkg/Mnt
Jul98	2.0	217	1,101	5.1
Mar99	2.1	297	1,559	5.2
Aug00	2.2	453	2,601	5.7
Jul02	3.0	859	5,119	6.0
Dec04	(3.1)	1237	7,786	6.3

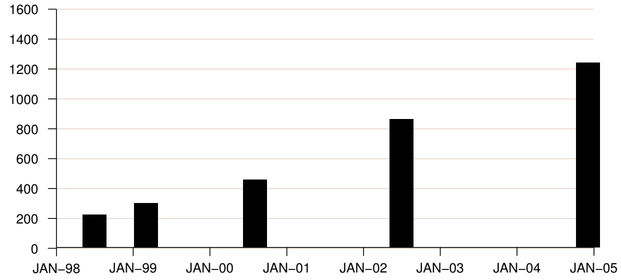

Fig. 2. On the left is shown the number of maintainers, packages and its ratio. On the right, the number of maintainers for each version at the time of its release.

Date	Release	Packages	Packages QA	Percentage
Jul 98	2.0	14	14	1.3%
Mar 99	2.1	21	11	1.4%
Aug 00	2.2	46	31	1.8%
Jul 02	3.0	101	71	2.2%
Dec 04	(3.1)	599	194	7.4%

TABLE II

MAINTAINER TEAMS FOR THE LAST FIVE STABLE DEBIAN RELEASES. THE COLUMN "PACKAGES" IS THE TOTAL NUMBER OF PACKAGES MAINTAINED BY MAINTAINED TEAMS (INCLUDING QA), THE COLUMN "PACKAGE QA" IS THE NUMBER MAINTAINED BY THE QA TEAM, "PERCENTAGE" IS OF PACKAGES MAINTAINED BY ALL TEAMS (INCLUDING QA)

by this very special team, the growth in the number of packages actually maintained by teams is clear.

This growth in the number of teams, along with the evolution of the packages per maintainer ratio leads us to the conclusion that the involvement of maintainers has been increasing with time.

C. Tracking remaining Debian Maintainers

At the time of the release of Debian 2.0 in July 1998 there were 216 voluntary maintainers contributing to Debian. We have studied how the involvement of these 216 contributors to Debian 2.0 changed over time. Table III gives an overview of the number of contributors from the original group left at each release, as well as the number of packages maintained by them. As the figure shows, the number decreases steadily, with only 121 of the original 216 contributors (55.8%) still contributing to Debian in December 2004. Hence, after six and a half years the half-life value has still not been achieved. If the current trend persists, that will happen at around 7.5 years (or 90 months). Taking other releases as the starting point gives similar values, so this seems to be the common trend. It would be interesting to perform further analysis about which factors influence how long volunteers remain active. There is already evidence that some volunteers face feelings of burn-out [12], but further studies into human-resource management and motivation in libre software projects would have positive effects on extending the half-life of volunteer contributions.

The number of packages for which these maintainers are responsible is also interesting. The initial number of packages maintained by the 216 contributors of Debian 2.0 was 1,101. The corresponding number for the maintainers remaining shrank to 729 for the last version considered. Again, the number of packages per maintainer has not decreased with time: the maintainers from the original set of July 1998 who were still active in December 2004 actually increased the number of packages they maintain.

Date	Release	Devs	Packages	Pkg/Dev
Jul 98	2.0	216	1,101	5.1
Mar 99	2.1	207	1,086	5.2
Aug 00	2.2	188	1,040	5.5
Jul 02	3.0	147	870	5.9
Dec 04	(3.1)	121	729	6.0

TABLE III

PACKAGES MAINTAINED BY THE DEBIAN 2.0 MAINTAINERS

D. Researching maintainer experience

In figure 3 it is shown when currently active maintainers got involved in the project. For every maintainer of a package in the latest release we have investigated in which release their first contribution can be found. In addition to the 121 maintainers who have made steady contributions since July 1998 (release 2.0), 55 participants got involved before Debian 2.1, and 114 arrived with Debian 2.2. In the last two stable releases, 393 and 554 new maintainers have been identified.

Although we had thought that more experienced maintainers would have a higher weight on their shoulders, the package per maintainer column in the table shows similar numbers. The most active maintainers are those who joined Debian towards the preparation of Debian 2.2. Further investigation is required to clarify why there is such a difference for that release.

E. Packages of maintainers who left the project

When maintainers leave the project, their packages become unmaintained (orphaned). These may be taken up by others, or will not be present in the next stable release, if they are not adopted. In table IV the ratios and numbers of orphaned and adopted packages between any pair of the studied releases are shown.

From it we have learned that the percentage of adopted packages is very high: more than 60% for all releases considered. This happens even for releases with a very high portion of orphaned packages (for instance, between 2.0 and 3.1). In other words, even though maintainers who left Debian between July 1996 and December 2004 were responsible for 33.5% of the packages in 2.0, 67.5% of these packages can still be found in 3.0. We can thus affirm that Debian counts on a natural ‘regeneration’ process for its voluntary contributors and that there is a high probability that the packages of a maintainer who leave the project will be adopted by others.

Another interesting fact is that the adopted to orphaned ratio is always decreasing for a given release. This means that the number of orphaned packages grows more quickly than that of adopted, i.e. there are some packages missing in every new release. Therefore, if a package is unmaintained and falls off the next release it will probably not enter a future one. In this study we have only considered removed packages from maintainers who left the project, but it is likely that some software will also be abandoned by maintainers still remaining active, and are therefore not covered by this study.

In any case, it should be noted that users are left unsupported when a package (maybe providing a unique functionality) from a previous release is not present in subsequent ones. Given this fact, it may be beneficial to establish mechanisms to ensure that packages which cannot be supported in the long term will not be introduced in the first place, or at least only in a section of the Debian repository which is clearly marked as being less supported.

Obviously, it is hard to determine in advance which packages are likely to be abandoned in the future. However, some parameters can be used to judge which projects are more likely to be sustainable. For example, the probability that a project which completely relies on the work of one developer remains active for five years is by far smaller than that of a bazaar-driven project with 30 core developers and an active community of contributors.

The key is to develop a framework that may assess projects regarding their sustainability which takes various factors into account, such as the number of developers, or the size of the user base. This assessment can also include simple security audits to establish whether a piece of software has sufficient quality so that the maintenance is possible. Even though no complete framework has been developed for the Debian project yet, there has been some work which can be adapted. For example, the Open Source Maturity Model is a step in that direction [13], [14].

F. Experience and importance

We have used data from the Debian Popularity Contest to find out whether more ‘important’ packages are maintained by more experienced volunteers. Table V shows the data corresponding to installations and use of packages by maintainers which are still in the project, and which were already present in the studied releases. In it we can see, for instance, how 2.0 and 3.1 have 121 common maintainers, which are responsible for 729 packages which have been installed 919,856 times and 362,249 are used regularly.

If we take the number of installations per maintainer and the number of regularly used packages per maintainer (‘Votes/Maint’) we can answer the question we proposed. According to our hypothesis these ratios decrease in time, which means that more experienced volunteers maintain packages which are installed and used more often. This can be observed through all Debian releases, with the exception of 2.2, which has higher values than those of 2.1. This complies with the former evidence from figure 3, where we saw that maintainers joining between 2.1 and 2.2 were particularly active. As was said, further investigation is needed to explain this difference in activity for those maintainers.

This is also an evidence that many of the essential components of the Debian system were introduced in the first releases, and that new packages are mostly add-ons and software that is not installed and used that often.

VI. Conclusions and further work

We have conducted a quantitative study of the evolution of the Debian maintainership over the last six and a half years. We have retrieved and analyzed publicly available data in order to find out how Debian handles the volatility of the volunteers who made it happen.

Some of the most interesting findings are:

- Both the number of Debian maintainers, and the number of packages per maintainer grow in time, even if there is a trend towards having maintainer teams.
- The number of maintainers from previous releases who remain active is very high, with an estimated half-live of around 7.5 years (90 months). More than half of the maintainers from Debian 2.0 still contribute to the current release.
- Developers tend to maintain more and more packages as they are more experienced in the project.
- However, this does not mean that maintainers who have been in the project for more time maintain more packages than newer maintainers. In fact, in the latest release the highest packages per maintainer ratio is

Date	Release	Devs	Packages	Pkg/Dev
Jul98	2.0	121	727	6.0
Mar99	2.1	55	338	6.1
Aug00	2.2	114	919	8.1
Jul02	3.0	393	2,544	6.5
Dec04	(3.1)	554	3,258	5.8

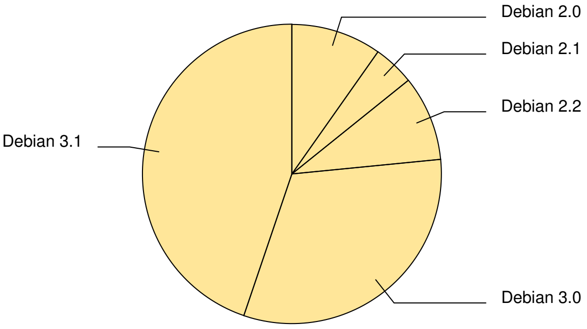

Fig. 3. First stable release for maintainers of packages present in Debian 3.1

Release 1	Release 2	Orphaned	Adopted	A/O	O/Tot1	O/Tot2
2.0	2.1	15	14	93.3%	1.3%	1.0%
2.0	2.2	61	40	65.6%	5.5%	1.5%
2.0	3.0	231	171	74.0%	21.0%	4.5%
2.0	(3.1)	372	251	67.5%	33.8%	3.2%
2.1	2.2	47	31	66.0%	3.0%	1.8%
2.1	3.0	302	220	72.8%	19.4%	5.9%
2.1	(3.1)	493	327	66.3%	31.6%	6.3%
2.2	3.0	281	207	73.7%	10.8%	5.5%
2.2	(3.1)	617	403	65.3%	23.7%	7.9%
3.0	(3.1)	596	383	64.3%	11.6%	7.6%

TABLE IV

ORPHANING AND ADOPTION OF PACKAGES. EACH ROW SHOWS PACKAGES PRESENT IN THE OLDER RELEASE (FIRST COLUMN) AND NOT IN THE NEWER ('ORPHANED' COLUMN), AND WHICH OF THOSE WERE ADOPTED. LAST COLUMNS SHOW THE PERCENTAGES OF PACKAGE 'SAVED' (ADOPTED TO ORPHANED, A/O), AND ORPHANED IN THE NEWER RELEASE TO TOTAL IN THE OLDER (O/TOT1) AND NEWER (O/TOT2) RELEASES.

Release	CMaint	CPkg	Installations	Votes	Inst/Maint	Votes/Maint
2.0	121	729	919,856	362,249	7602.1	2993.8
2.1	176	1,066	1,306,067	498,061	7420.8	2829.9
2.2	290	1,984	2,135,137	805,642	7362.5	2778.0
3.0	683	4,528	3,712,435	1,280,173	5435.9	1874.3
(3.1)	1237	7,786	4,566,601	1,487,246	3691.7	1202.3

TABLE V

INSTALLATIONS AND REGULAR USE OF PACKAGES. THE CMAINT COLUMN SHOWS HOW MANY MAINTAINERS 3.1 HAD IN COMMON WITH THE RELEASE IN THE FIRST COLUMN, WHILE THE CPKG SHOWS THE NUMBER OF PACKAGES MAINTAINED BY THEM. COLUMNS INSTALLATIONS AND VOTES GIVE THE SUM OF THE PACKAGES INSTALLED AND VOTED (USED REGULARLY) FOR THOSE PACKAGES MAINTAINED BY COMMON MAINTAINERS. THE LAST TWO COLUMNS SHOW THE RATIOS OF BOTH TO COMMON MAINTAINERS.

shown by those entering the project around the year 2000.

From these facts, it can be said that Debian maintainers tend to commit to the project for long periods of time. However, there is a worrisome trend towards a higher and higher ration of packages per maintainer, which could imply scalability problems as the number of packages in the distribution increases, if the project doesn't admit a proportional number of developers.

Another issue on which we have focused is what happens to those packages that were maintained by developers who left the project. Most of them are taken over by other maintainers so that we can state that a natural 'regeneration' exists. Based on the data we have researched, those packages which are not adopted by other maintainers in the next release, and are therefore not present in it, are unlikely to be re-introduced in future releases.

Finally, we have also found that more experienced maintainers are responsible for packages which are installed and used regularly more often.

In addition to the new insights gained in this investigation, we have proposed a number of further studies to elaborate the findings of the present paper. In particular, team maintenance and its impact on the quality of packages would be interesting to research. It is also not clear why there is an increase in the ratio of packages per maintainer. Possible explanations are that better tools and practices lead to more efficiency or that with the success of libre software new volunteers show more motivation and commitment, but more data is needed before these explanations can be conclusive.

From a more general point of view, this study explores the behavior of volunteers in libre software projects, and provides some answers to why this kind of voluntary contributions are capable of producing such large, mature and stable systems over time, even when the project has no means for forcing any single developer to do any given task or may leave the project during important development phases. It is impossible to infer the behavior of volunteer developers just from the study of a single project, but given the size and relevance of the Debian project, at least some conclusions can be exposed as hypothesis for validating in later research efforts.

One of them is the stability of volunteer work over time. The mean life of volunteers in the project is probably larger than in many software companies, which would have a clear impact on the maintenance of the software (it would be likely that developers with experience in a module be available for its maintenance over long periods of time). Another one is that volunteers tend to take over more work with the passing of time if they manage to stay in the project: in other words, they voluntarily increase their responsibilities in the project. Whether this is because it is easier for them because of their experience, or because they devote more effort to the project, is for now an open question. Yet a third one is the stability of the voluntary effort when some individuals leave the project: most of their work is taken over by other developers. Therefore, despite being completely based on volunteers, the project organizes itself rather well with respect to leavings, which is an interesting lesson about how the project can survive in the long term.

As a final summary, we have found that given that there are no formal ways of forcing a developer to assume any given task, voluntary efforts seem to be more stable over time, and more reliable with respect to individuals leaving the project than we had expected in advance.

VII. Acknowledgements

The work of Gregorio Robles and Jesus M. Gonzalez-Barahona has been funded in part by the European Commission under the CALIBRE CA, IST program, contract number 004337, in part by the Universidad Rey Juan Carlos under project PPR-2004-42 and in part by the Spanish CICyT under project TIN2004-07296. The work of Martin Michlmayr has been funded in part by Fotango, the NUUG Foundation and the EPSRC. We also want to thank the anonymous reviewers for their extensive comments.

References

[1] D. M. German, “Decentralized Open Source global software development, the GNOME experience,” *Journal of Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 201–215, 2004.

[2] M. Monga, “From Bazaar to Kibbutz: How freedom deals with coherence in the Debian project,” in *Proc 4th Workshop on Open Source Software Engineering*, Edinburg, UK, 2004.

[3] S. O’Mahony, “Guarding the commons: how community managed software projects to protect their work,” *Research Policy*, no. 32, pp. 1179–1198, 2003.

[4] “Debian Social Contract,” http://www.debian.org/social_contract.

[5] G. Garzarelli and R. Galoppini, “Capability coordination in modular organization: Voluntary FS/OSS production and the case of Debian GNU/Linux,” November 2003.

[6] M. Michlmayr and B. M. Hill, “Quality and the reliance on individuals in free software projects,” in *Proc 3rd Workshop on Open Source Software Engineering*, Portland, USA, 2003, pp. 105–109. [Online]. Available: http://www.cyrius.com/publications/michlmayr_hill-reliance.pdf

[7] M. Michlmayr, “Managing volunteer activity in free software projects,” in *Proc USENIX 2004 Annual Technical Conference, FREENIX Track*, Boston, USA, 2004, pp. 93–102. [Online]. Available: http://www.cyrius.com/publications/michlmayr-mia.pdf

[8] J. M. González-Barahona, G. Robles, M. Ortuño Pérez, L. Rodero-Merino, J. Centeno González, V. Matellan-Olivera, E. Castro-Barbero, and P. de-las Heras-Quirós, “Analyzing the anatomy of GNU/Linux distributions: methodology and case studies (Red Hat and Debian),” in *Free/Open Source Software Development*, S. Koch, Ed. Hershey, PA, USA: Idea Group Publishing, 2004, pp. 27–58.

[9] M. Lehman, J. Ramil, P. Wernick, and D. Perry, “Metrics and laws of software evolution - the nineties view,” in *Proceedings of the Fourth International Software Metrics Symposium*, Portland, Oregon, 1997.

[10] C. Lameter, “Debian GNU/Linux: The past, the present and the future,”
http://telemetrybox.org/tokyo/, 2002.

[11] J. M. Gonzalez-Barahona, M. A. Ortuño Perez, P. de las Heras Quiros, J. Centeno Gonzalez, and V. Matellan Olivera, “Counting potatoes: the size of Debian 2.2,” *Upgrade Magazine*, vol. II, no. 6, pp. 60–66, Dec. 2001.

[12] G. Hertel, S. Niedner, and S. Herrmann, “Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel,” *Research Policy*, vol. 32, no. 7, pp. 1159–1177, 2003.

[13] “The Open Source maturity model: A methology for assessing open source software,”
http://www.navicasoft.com/pages/osmm.htm.

[14] B. Golden, *Succeeding with Open Source*. Addison-Wesley Professional, 2004.