Geographic Location of Developers at SourceForge*

Gregorio Robles
grex@gsyc.escet.urjc.es

Jesus M. Gonzalez-Barahona
jgb@gsyc.escet.urjc.es

Grupo de Sistemas y Comunicaciones
Universidad Rey Juan Carlos
Mostoles, Spain

ABSTRACT

The development of libre (free/open source) software is usually performed by geographically distributed teams. Participation in most cases is voluntary, sometimes sporadic, and often not framed by a pre-defined management structure. This means that anybody can contribute, and in principle no national origin has advantages over others, except for the differences in availability and quality of Internet connections and language. However, differences in participation across regions do exist, although there are little studies about them. In this paper we present some data which can be the basis for some of those studies. We have taken the database of users registered at SourceForge, the largest libre software development web-based platform, and have inferred their geographical locations. For this, we have applied several techniques and heuristics on the available data (mainly e-mail addresses and time zones), which are presented and discussed in detail. The results show a snapshot of the regional distribution of SourceForge users, which may be a good proxy of the actual distribution of libre software developers. In addition, the methodology may be of interest for similar studies in other domains, when the available data is similar (as is the case of mailing lists related to software projects).

Categories and Subject Descriptors

D.2.m [**Software Engineering**]: Miscellaneous

General Terms

Human Factors

*This work has been funded in part by the European Commission, under the CALIBRE CA, IST program, contract number 004337 and under the FLOSSWorld SA, IST program, contract number 015722. This work is based on the SourceForge database provided by University of Notre Dame, see details at http://www.nd.edu/ oss/Data/data.html.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
MSR'06, May 22–23, 2006, Shanghai, China.
Copyright 2006 ACM 1-59593-085-X/06/0005 ...$5.00.

Keywords

Geographical location, mining software repositories, libre software, free software, open source software

1. INTRODUCTION

One of the most well known characteristics of libre (free, open source) software[1] is the worldwide distributed pool of developers that collaborate in tens of thousands of projects, using Internet-based tools for coordination. These projects are usually open to participation by anyone, from any corner of the globe, provided Internet access is granted; those with enough knowledge and skills can, in principle, join them. This openness, and the underlying informality, has resulted in an environment where participation is difficult to control, or even understand. One of the most significative examples of open issues in this respect is the geographical distribution of the aforementioned pool of developers. The answer to the question "where do developers live?" is not only interesting for academic reasons; it is also important from both strategic and economic points of view.

In this paper, we present a first approach to deal with this question by analyzing data about a huge sample of developers. We describe how we have mined the database of the largest libre software development supporting platform (SourceForge) looking for indicators to estimate the geographic location of the developers registered in it. Since the number of users of the SourceForge platform is well over one million, we can assume it is a reasonably good and representative proxy of the whole population of libre software developers (although for sure it presents some bias, as will be discussed later, for instance in terms of language knowledge).

The main goals of this paper are two: to show a methodology to estimate country of residence (as a simple quantifier of geographic location) using the indicators available in the SourceForge database, and to obtain a first estimation of the location of libre software developers.

With respect to the first goal, it is noteworthy to mention that SourceForge does not store specific information about the geographical location of developers, which therefore has to be inferred from other indicators, such as the domains in the e-mail address, or the time zone information developers introduce when registering at SourceForge. We believe that

[1]Through this paper we will use the term "libre software" to refer to any code that conforms either to the definition of "free software" (according to the Free Software Foundation) or "open source software" (according to the Open Source Initiative).

the methodology we have designed for this inference can be extended to deal with data from other sources, such as mailing lists.

With respect to the second goal, our estimation will be only as precise as SourceForge population is representative of the global libre software development population. We offer no proof of this representativeness, and therefore the results presented have to be considered with care. However, despite any bias the SourceForge population can have, it is the most global, diverse and (by far) largest community of libre software developers, which means that, even if the results were not extensible to the whole development community, they are interesting by themselves.

The structure of this paper is as follows. In the next section we present some other research efforts on the geographical distribution of libre software developers. Afterwards, while the third section contains a description of the data source we have used for the study, the forth one presents the methodology that we have designed to infer the nationalities. Next, the results of the application of the methodology to the SourceForge population is shown and briefly commented. Finally, conclusions and some ideas for further research are be offered in the last section.

2. RELATED RESEARCH

Among the several approaches to study the geographical location of libre software developers, we can identify two categories, according to the data acquisition process: those which collect specific data provided by certain libre software projects (such as the CREDITS files found with the source code, or information available on the web pages of the project), and those which obtain the data by surveying developers.

To our knowledge, the first study in this field [3] studied the meta-data that can be found in the Linux Software Map entries[2]. Among other fields, they contain the name and e-mail address of the main author. By studying the top-level domain[3] of the e-mail address, the country of residence could be partly inferred, although the presence of generic top-level domains[4] made it impossible to determine the location of many developers, especially those based in the United States. Hence, there is a bias, recognized by the authors, which recommend further research on this matter.

The Debian project was studied in 2001 [12], based on the country information that the Debian developers introduce in the Debian Developer Database. Since it also contains information about the admission date for each developer, an evolutionary analysis was performed, showing how Debian started primarily as an US-based project, turning later to an European majority. The presence of members of developing countries was minimal.

The CREDITS file of the Linux kernel, and the contact information of the GNOME project was also studied in 2001 [6]. Its most remarkable result is that the shift towards a more European-based development in both projects can be explained by economic theory, with the number (and distribution) of developers depending on the cost of opportunity. Some years later, a new study of the Linux CREDITS file [13] provided a more in-depth study of the geographical distribution of the kernel developers.

One of the first studies based on surveys was WIDI [12] (2001) which featured over 5,500 respondents. Results showed a majority of EU-based developers, although the self-selected nature of the participants introduced a bias which has to be taken into account. A later survey, FLOSS [4], was answered by about 2,500 self-selected developers over the Internet. Although it did not include the study of the geographical distribution, a surprisingly large quantity of European developers (in comparison with their American and Asian counterparts) participated. This was one of the reasons to perform similar survey with other *flavors*, such as FLOSS-US [2] (interestingly enough, Europeans where also predominant) and other Asian surveys.

Regarding SourceForge, it has been an inspiration for many research papers on libre software and software repositories in general. The most relevant to our work is maybe a statistical analysis of the projects hosted in SourceForge [5], which shows that it hosts many small to medium-sized projects, while larger ones (such as Linux, GNOME, KDE or Apache) tend to use their own development infrastructure. For our purposes, this is by no means a disadvantage, since many developers who contribute to large projects are also registered at SourceForge. We can, hence, consider SourceForge users population as the largest collection of libre software developers in the world.

3. DATA SOURCES

The data source analyzed in this work is the SourceForge database, as provided to research teams by the University of Notre Dame. The database is provided as a monthly dump under an special agreement[5]. Therefore, the data set we use is not public, but is available to the research community, which means that the results based on it are reproducible by other groups.

For our research, we use the *private* e-mail address and the time zone associated to every SourceForge user in the database. SourceForge uses the private e-mail address for verification purposes. It is private in the sense that it is not published in the site. The time zone can be specified by registered users, in which case it is used to localize the display of time when the user is logged in. Usually time zones contain the region and a city name (eg. Europe/Madrid), although there are other formats, such as abbreviations (eg. CET is Central European Time)[6]. The default choice, for users which have not selected a time zone, is GMT.

The SourceForge data is provided through a web-based tool. Queries on it are returned in a text file, with the database fields separated by semicolons. We have queried for the private e-mail and time zone fields, parsed the output, transformed it into and SQL dump, and fed a database with the data. After this process, we hold data for more than 1,180,000 registered users at SourceForge in November 2005.

[2]The Linux Software Map (LSM) is a database of software written or ported to Linux, `http://lsm.execpc.com/lsm/`.

[3]A top-level domain (TLD) is the last part of an Internet domain name; that is, the letters which follow the final 'dot' of any URL.

[4]A generic top-level domain (gTLD) is in theory used for a particular class of organizations (com for commercial organizations, edu for educational institutions, etc.). Those domains do not include geographic information.

[5]More information about this agreement can be obtained from http://www.nd.edu/ oss/Data/data.html

[6]For a complete list of time zones, visit: http://wwp.greenwichmeantime.com/info/timezone.htm.

This is by far the largest data set ever used to estimate the geographical distribution of libre software developers.

Another data set based on SourceForge, FLOSSMole [1], provides public information about its registered users, but only includes the information that can be retrieved from the public interface of the site. Therefore, it does not include the private e-mail address, which is basic for this study.

4. METHODOLOGY

The final goal of the methodology described in this section is to estimate, as accurately as possible, the geographical distribution of the users in the database, using the domain in their e-mail address and the time zone as the base for the analysis.

The inference is straightforward when the TLD (top-level domain) of the e-mail address corresponds with a country code (country-coded top-level domains, or ccTLD; table 1 displays a list with some ccTLDs and the country they are assigned to). This is for instance the case of one of the authors of this paper, who is registered at SourceForge with following e-mail address: *gsyc.escet.urjc.es* ('.es' is the ccTLD corresponding to Spain).

ccTLD	Country
de	Germany
es	Spain
fr	France
mx	Mexico
uk	United Kingdom
us	United States

Table 1: List of some country coded top-level domains (ccTLDs).

The same can be said for many time zone codes. Almost all countries have a time zone designated by region, or even an abbreviated one. For instance, the Europe/Madrid timezone is a good indicator about the developer being located in Spain. As in the case with ccTLDs, it is trivial to assign a time zone to a country (and therefore to a ccTLD). As a matter of example, table 2 displays some time zones and their corresponding ccTLDs.

Time zone	ccTLD
Europe/Berlin	de
US/Eastern	us
America/New_York	us
EST	us
Europe/Madrid	es
Europe/Moscow	ru
America/Sao_Paulo	br

Table 2: List of some time zones and country codes top-level domains (ccTLDs).

However, in many cases the identification of the country of origin is not that simple. The reason for this is basically because we have to face incomplete information.

4.1 Incomplete information

Unfortunately, from our total population of over 1,180,000 registered developers, more than 750,000 do not use a ccTLDs (67%) in their e-mail address. The use of generic top-level domains (gTLD), such as .com, .net, .org, .biz, .info, is widespread, and renders the identification of national origin more difficult.

With respect to time zones, around 425,000 have the default, GMT (38%). This is problematic, since in this specific case we cannot assume that the time zone was selected: maybe the user lives in any other time zone, but never set it, or maybe she lives in a GMT time zone (and therefore have correctly selected it). However, they should still be assigned to some national origin (since there are several countries with GMT time).

Fortunately, we can build upon the fact that we have both entries for all registered users, and one of them can be enough to have evidence about the country. This means that the 'problematic' records are only those that have a gTLD in the e-mail address and GMT time zone. There are about 280,000 users (25% of the total population) in this situation. Our aim in this section is to find ways to lower the percentage of users to which we cannot assign a country. Several methods will be used in this sense. We will start by inferring information from the second level gTLDs (SLDs).

Domain	Number
hotmail.com	63784
yahoo.com	40180
gmail.com	14191
aol.com	6275
gmx.net	4128
msn.com	3688
163.com	2013
ntlworld.com	1998
rr.com	1981
rediffmail.com	1881

Table 3: Top 10 domains in number of SourceForge users that have set GMT as their time zone (total: 66054 distinct domains).

Table 3 gives the top ten SLDs in number of developers with GMT as time zone. For the SLDs with many registered users, we can look for those who specified a time zone different from GMT, and, in a first approach, assume that users who specified GMT should have the same proportion of non-GMT time zones. In other words, we propose an algorithm to proportionally distribute those users with a GMT time zone among all other time zones found for a SLD (see figure 1 for a graphical display of this idea). So, the algorithm takes those SLDs with a gTLD and finds out the time zone that the corresponding users selected (from which we can infer the country). Then it assigns proportionally entries with GMT time zone to the given countries.

As a case of example, consider epo.org, a domain with 22 registered users. From these, 10 had set GMT as time zone, 8 had the Dutch Europe/Amsterdam, 2 had the German Europe/Berlin, 1 the Austrian Europe/Vienna and a last one the French Europe/Paris time zone.

The algorithm in this first approach would assume that the GMT entries have to be assigned proportionally to the other ones. This means that there are 10 entries to be split among the other countries. To make it proportionally, all non-GMT time zones are added up. The final estimation for each country is given by the sum of the original number of developers plus the proportional part of the GMT. Values

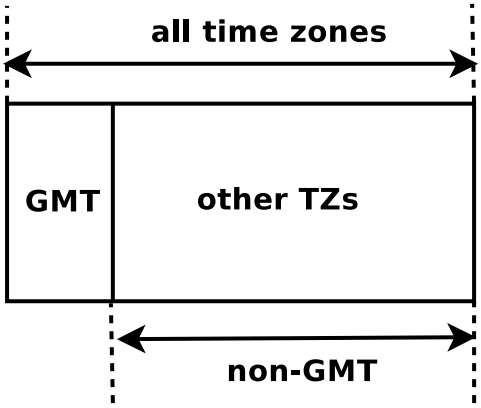

Figure 1: Redistribution algorithm graphically.

are not rounded at this point as this algorithm is going to be applied on all other gTLDs with GMT entries, so for the sake of accuracy rounding should occur at the end of the process. The following excerpt should clarify the algorithm:

```
GMT: 10 <---- to be distributed
nl:   8 -> 8 + 8/12 * 10 =  14.67 <- nl
de:   2 -> 2 + 2/12 * 10 =   3.67 <- de
at:   1 -> 1 + 1/12 * 10 =   1.83 <- at
fr:   1 -> 1 + 1/12 * 10 =   1.83 <- fr
                           -------
                  Total:    22
```

However, this algorithm presents problems for those countries which are actually in GMT, by underestimating their number of developers. Next subsection explains why, and shows a second approach which solves this problem. For the final estimation, this second approach will be used.

4.2 Countries in the GMT zone

The previous algorithm has a problem for those countries located in the the GMT zone, which in our data set are mainly the United Kingdom (uk), Ireland (ie) and Portugal (pt), since they are underrepresented. This is because we have assumed that those users who selected the GMT time zone did it 'by error' (never changing the default value). This is not always true, so we should find ways, to compensate this effect.

The basic idea for the subsequent reasoning is the assumption that those who live in the GMT time zone behave the same when filling out their data than the rest of the population. In other words, the 'error' rate of leaving the default time zone would be similar for all entries. Table 4 shows, for many European countries, the number of users with their 'own time zone' (time zone that corresponds to their respective ccTLD), and those that have selected GMT.

For instance, from those who have an Austrian (at) TLD, 3229 have chosen Europe/Vienna as their time zone, while 2840 left the default GMT. The last column shows the ratio between the own time zone and GMT. It is clear that these ratios are completely different for those countries that lay within the GMT time zone (with values below 0.3), when compared to the rest of European countries (with values in general between 1.10 and 1.90).

Country	own TZ	GMT	Ratio
at	3229	2840	1.14
be	4256	2701	1.58
ch	3813	2864	1.33
cz	2999	1708	1.76
de	36471	30857	1.18
dk	3779	2362	1.60
es	3930	2699	1.46
fi	3087	1187	2.60
fr	12150	8847	1.37
gr	1339	687	1.95
hu	2976	1957	1.52
it	12556	8917	1.41
lu	162	117	1.38
nl	9483	6027	1.57
no	2546	1620	1.57
pl	7607	4403	1.73
se	5817	3061	1.90
Total	116200	82854	1.40
ie	89	996	0.09
pt	632	2514	0.25
uk	2854	22108	0.13

Table 4: Time zone choice for some European countries.

If we take all European countries, the weighted mean of the ratio between the own time zone and GMT is 1.40. It is reasonable to assume that United Kingdom (uk), Ireland (ie) and Portugal (pt) should have a similar mean for that ratio. This assumption makes it possible to find a factor that can be multiplied to the entries corresponding to these countries in the GMT-assignation algorithm explained in the previous subsection. The equation for calculating this factor is:

$$Factor = \frac{GMT + ownTimezone}{1.71 * ownTimezone} \quad (1)$$

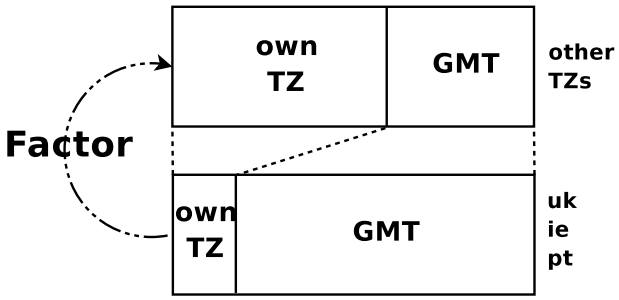

Figure 2: GMT factor calculation graphically.

How is that equation obtained? As shown in figure 2, Factor should ensure that the ratio of GMT to own time zone, for the GMT countries, is similar to that of non-GMT European countries. Therefore, we can define some conditions that must be met. As shown by equation 2, the number of entries before (given by the C, for current, subindex) and after (given by the F, for final, subindex), considering the factor, should remain constant.

$$GMT_F + ownTimezone_F = GMT_C + ownTimezone_C \quad (2)$$

A second condition is that the ratio between the final GMT and the final own time zone entries has to be 1.4, since this is the weighted mean of the ratio between the own time zone and GMT for the European countries (see equation 3):

$$1.40 * GMT_F = ownTimezone_F \quad (3)$$

A third condition introduces Factor (see equation 4), stating that the number of entries given for the final own time zone has to be the same found for the current one multiplied by the factor (i.e. the number of users remains constant).

$$ownTimezone_F = Factor * ownTimezone_C \quad (4)$$

Given these three conditions, GMT_F, $ownTimezone_F$ and $Factor$ are the unknown parameters. If we solve the systems of equations, we get equation 1, which shows how the factor is calculated.

The factors obtained for the GMT countries can be found in table 5. These values mean, for instance, that every *uk* entry for a domain should be weighted as 5.1 entries from other non-GMT countries when performing the redistribution algorithm presented above (and depicted in figure 1).

Country	Factor
uk	5.1
ie	7.1
pt	2.9

Table 5: Multiplying factor for GMT countries.

Finally, there is a small set of users (around 3% of the total sample) that have GMT as time zone, hold an e-mail address with a gTLD and do not share SLD with any other SourceForge user. For this set of developers we obtain the IP of the SLD by querying a DNS server. Using the geoIP library[7] we query for the geographical location of the host. The geoIP library contains a database that maps IPs to countries. This method is used, for instance, to assign a developer with the *hautpraxis.com* SLD to Germany (de).

4.3 Inferring geographical location

We have described several ways of obtaining the country of residence of our developer base. It can be done by looking at the TLD of the e-mail address, by transforming the time zone, by assigning the time zone proportionally from those given by the ones who share SLD, and if none of these are possible, by looking the geographical information of the SLD. All this means that we may have various information sources for a given developer and that information may be fragmented.

Figure 3 displays the four sets that we can find in our sample: ccTLD-other is the set of developers having an e-mail address with a ccTLD and a time zone different from GMT. ccTLD-other includes those with a ccTLD e-mail address and GMT as time zone (probably some of them will actually live in a GMT time zone, but many others just left the default). Those developers with a gTLD address and a non-GMT time zone are grouped in gTLD-other. Finally, gTLD-GMT contains those with a gTLD and GMT.

As we have seen in this paper, depending on the zone we can obtain the country of the developers by different means; sometimes by more than one for each set. For developers in ccTLD-other we could assign a country based on the ccTLD (method ccTLD) or from the time zone (method TZ). In the case of ccTLD-GMT, the assignation could be by studying the ccTLD (method ccTLD) or by redistributing the GMT time zone among the rest of time zones (method GMT-redist). The only possibility for those in the C set is to obtain the country from the time zone, while for D we can get it by redistributing the time zone among the SLDs. For those in D for which this is not possible (the ones who have set GMT and do not share a SLD with other SourceForge user that makes redistribution possible), the IP address of the SLD can be taken into account.

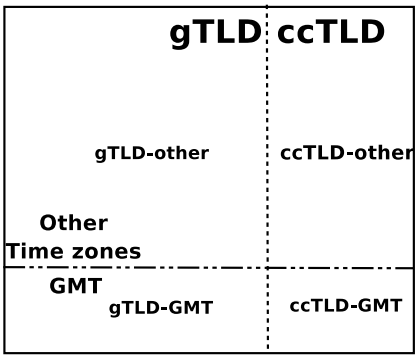

Figure 3: Set diagram with the different kinds of data.

Therefore, for ccTLD-other and ccTLD-GMT we have several choices. The yet unresolved question is to know which of them is better, since we do not know whether information provided by ccTLDs is more or less accurate than that obtained from the time zone.

5. RESULTS AND OBSERVATIONS

The results shown below have been obtained by assigning the ccTLD (so, we chose method ccTLD). We have also calculated results using method TZ (time zone instead of ccTLD) and GMT-redist (time zone distribution instead of ccTLD) and they are not significantly different.

Table 6 lists the top 50 countries by number of developers registered at SourceForge. These countries amount for 96.5% of the total identified registered users, while the top 20 countries include up to 83.9% of the total SourceForge population.

Although it is outside the scope of this paper, it would be interesting to find correlations between these data and some other per-country parameters, such as GDP or percentage of homes with Internet access. Just as a quick note, it is interesting to notice that this list is quite similar to the top countries by GDP, with some notable exceptions, of which the place of Japan (second by GDP) is probably the most surprising. Canada and Australia, on the other hand, are well above their ranking by GDP.

[7] `http://sourceforge.net/projects/geoip/`

Rank	Country	Developers
1.	United States	425620
2.	Germany	95800
3.	United Kingdom	60768
4.	Canada	49109
5.	France	44587
6.	China	36517
7.	Australia	31812
8.	Italy	30763
9.	Netherlands	29335
10.	Sweden	23867
11.	India	22113
12.	Brazil	21291
13.	Russian Federation	19012
14.	Spain	18905
15.	Japan	15081
16.	Poland	14697
17.	Belgium	13983
18.	Switzerland	12133
19.	Austria	10024
20.	Denmark	9952
21.	Singapore	9155
22.	Finland	9027
23.	Norway	8498
24.	Mexico	8185
25.	South Korea	7727
26.	Israel	6948
27.	Argentina	6695
28.	Hungary	6573
29.	Romania	6345
30.	Taiwan	6336
31.	Turkey	6099
32.	Czech Republic	6039
33.	European Union	5801
34.	South Africa	5706
35.	Portugal	4991
36.	New Zealand	4518
37.	Greece	4058
38.	Indonesia	3893
39.	Thailand	3746
40.	Bulgaria	3606
41.	Ukraine	3383
42.	Malaysia	3189
43.	Western Samoa	2856
44.	Ireland	2686
45.	Chile	2548
46.	Slovak Republic	2141
47.	Maldives	2067
48.	Colombia	2052
49.	Madagascar	1838
50.	Estonia	1758

Table 6: Results for the top 50 countries.

Also noteworthy is that we can find European Union in place 33 in the country list. This is because of the existence of the .eu domain and the CET time zone that could be assigned to a wide range of European countries (mostly part of the European Union). A way to address this problem could be using a redistribution algorithm for those entries, distributing proportionally among countries.

Surprising positions are achieved by Western Samoa (43), Maldives (47) and Madagascar (49). The reason for this is that some ccTLDs can be acquired without restrictions and have become *de facto* gTLDs. For instance, Western Samoa's top-level domain is ws, which has been sold as a shortcut for "website". Again, redistributing these entries in any of the ways already presented would make results more accurate. A good way of identifying these inflated domains would by correlating our results with total population, thus obtaining a per capita distribution. Per capita values that are too high will be a clear indicative.

Region	Developers
Africa	12560
Asia	127275
EU	401845
Europe	466792
North America	485679
Oceania	46422
South America	36330

Table 7: Results by regions.

Table 7 groups countries by regions. These figures are consistent with previous studies, maybe showing higher numbers for North America. In any case, it is clear that most of the developers come from Europe and North America (on an almost 50-50 ratio), followed by Asia with less than 10%. On the other hand, as the population is larger in Europe than in North America, this means that the *penetration* of the libre software development measured in SourceForge registered developers per capita is higher in North America than in Europe.

All of these results are of course not exact. We have worked with sources with rather different error margins, and we have used heuristics that are sound, but have for sure a certain error rate. To assess on the validity of the methodology for estimating the national origin, we should check (probably by contacting developers themselves) for a large fraction of SourceForge users. The results should then be compared with those of our study. However, the validations we have performed seem to indicate that the results are statistically sound, and that the figures shown are at least good estimators of the reality.

6. CONCLUSIONS AND FURTHER RESEARCH

In this paper we have described the process of extracting data about national origin from the SourceForge database, using mainly two parameters: e-mail addresses and time zones. We have also presented and discussed the results of applying this process to well over one million of registered users.

We have described the methodology with as much detail as possible, so that it can be completely understood and applied by third parties to this and other data sources. For instance, many methods described here can be used in other contexts, such as the study of contributions to the mailing lists archives of a project (provided there is access to the archives of those mailing lists).

Our methodology is not focused on identifying the geographical location of single developers (although in many cases that is done), but on finding the aggregate numbers of developers of a certain national origin. Therefore, we use in

many cases statistical relationships to infer the proportion of nationals of a certain country in a population of users with some characteristics. This is certainly a limitation of the proposed approach, specially if we were interested in (individual) developer identification methods as proposed in other works [10].

A future line of research could be to relate our findings with the activity of developers in the projects they are involved. This could be done by tracking developers in control versioning systems, mailing lists, forums, etc., and studying their activity by national origin. This could be an important issue, since previous research has shown that activity in libre software tends to be highly skewed towards a minority group responsible for the vast majority of the work performed. The authors of this work have started to analyze the CVS versioning system logs of all the SourceForge projects with the CVSAnalY tool [11], and the FLOSSMole project [1] has also information related to projects in the site. Both data sets could be used for this matter.

An interesting issue is how representative this study is of the whole population of libre software developers. SourceForge is not the only development platform: large libre software projects usually administrate their own infrastructure, and also many other SourceForge-like sites exist, in some cases linked to language or national communities. This means on one hand that we are not considering a lot of libre software which is being developed outside SourceForge (although many of the developers of that software are probably also users of this site), and on the other that the study could be skewed by ignoring some communities which are not represented in SourceForge, but in other facilities. Further studies should address this issue, and determine how good the SourceForge population is as a proxy of the developer population.

On a more socio-economic perspective, the findings presented in this paper could be related to other parameters characterizing the countries, looking for correlations which could explain the different quantities of developers, such as the GDP, the GDP per capita, Nielsen/Netratings, or other economic and technological parameters.

Especially interesting is also the issue of finding projects that are driven by local activity, i.e. projects whose contributors are from the same country, region or cultural environment. This could be a way of finding possible splits of the libre software community, and a first step towards identifying parameters leading to collaboration between developers. Cultural, language and other barriers should also be considered. In this sense, a recent change in the SourceForge platform has been the inclusion of a language field (although up to the moment less than 25% have specified a different language from the default 'English').

All of this could also be extended to a social network analysis, such as performed on libre software developers [8, 7, 9], but taking into account geographical information.

7. ACKNOWLEDGMENTS

We thank the SourceForge team, and Greg Madey from the University of Notre Dame, for providing access to the SourceForge data. Also, a big thank you goes to our colleagues from GSyC/LibreSoft for their help verifying the validity of the data.

8. REFERENCES

[1] M. Conklin, J. Howison, and K. Crowston. Collaboration using OSSmole: A repository of FLOSS data and analyses. In *Proceedings of the International Workshop on Mining Software Repositories*, pages 126-130, St. Louis, Missouri, USA, May 2005.

[2] P. A. David, A. Waterman, and S. Arora. FLOSS-US. The Free/Libre/Open Source Software Survey for 2003. *Technical report*, Stanford Institute for Economic and Policy Research, Stanford, USA, 2003.

[3] B. J. Dempsey, D. Weiss, P. Jones, and J. Greenberg. A quantitative profile of a community of Open Source Linux developers. *Technical report*, October 1999.

[4] R. A. Ghosh, R. Glott, B. Krieger, and G. Robles. Survey of developers (free/libre and open source software: Survey and study). *Technical report*, International Institute of Infonomics. University of Maastricht, The Netherlands, June 2002.

[5] K. Healy and A. Schussman. The ecology of open-source software development. *Technical report*, University of Arizona, USA, Jan. 2003.

[6] D. Lancashire. Code, culture and cash: The fading altruism of Open Source development. *First Monday*, 6(12), 2001.

[7] L. Lopez, J. M. Gonzalez-Barahona, and G. Robles. Applying social network analysis to the information in CVS repositories. In *Proc Intl Workshop on Mining Software Repositories*, pages 101-105, Edinburg, UK, 2004.

[8] G. Madey, V. Freeh, and R. Tynan. The open source development phenomenon: An analysis based on social network theory. In *Americas Conf on Information Systems*, pages 1806-1813, Dallas, TX, USA, 2002.

[9] M. Ohira, N. Ohsugi, T. Ohoka, and K.-I. Matsumoto. Accelerating cross-project knowledge collaboration using collaborative filtering and social networks. In *Proceedings Intl Workshop on Mining Software Repositories*, St. Louis, Missouri, USA, May 2005.

[10] G. Robles and J. M. Gonzalez-Barahona. Developer identification methods for integrated data from various sources. In *Proceedings of the International Workshop on Mining Software Repositories*, pages 106-110, St. Louis, Missouri, USA, May 2005.

[11] G. Robles, S. Koch, and J. M. Gonzalez-Barahona. Remote analysis and measurement of libre software systems by means of the CVSAnalY tool. In *Proc 2nd Workshop on Remote Analysis and Measurement of Software Systems*, pages 51-56, Edinburg, UK, 2004.

[12] G. Robles, H. Scheider, I. Tretkowski, and N. Weber. Who is doing it? A research on libre software developers. *Technical report*, Technische Universitaet Berlin, Berlin, Germany, Aug. 2001.

[13] I. Tuomi. Evolution of the Linux Credits file: Methodological challenges and reference data for Open Source research. *First Monday*, 9(6), 2004.