Mining for Localization in Android

Laura Arjona Reina, Gregorio Robles
Universidad Rey Juan Carlos
Madrid, Spain
larjona99@gmail.com, grex@gsyc.urjc.es

***Abstract*—Localization, and in particular translation, is a key aspect of modern end-user software applications. Open source systems have traditionally taken advantage of distributed and volunteer collaboration to carry localization tasks. In this paper, we will analyze the Android source code repository to know how localization and translation is managed: who participates in this kind of tasks, if the translation workflows, participants and processes follow the same patterns as the rest of the development, and if the Android project takes benefit from external contributions. Our results show that Android should ease the localization tasks to benefit from external contributions. Steps towards obtaining a specialized team as found in many other free software projects are also encouraged.**

***Keywords*-Android; translation; localization; i18n; l10n; mining software repositories;**

This is a preprint version of the paper to be found in: “2012 9th IEEE Working Conference on Mining Software Repositories (MSR)”
Digital Object Identifier: 10.1109/MSR.2012.6224272

I. Introduction

Internationalization (i18n in its short form) is the process of designing software so that it can be adapted to different languages, regions or target environments. Localization (l10n in its short form) is the process of adapting the software to a specific locale or target environment, for example translating texts to a specific language. Both are key aspects for software dissemination. Opening a project for volunteer contributions in localization tasks dramatically increases the number of languages to which a certain piece of software can be successfully translated, since non-technical users can also participate thanks to i18n efforts to separate the locale-specific strings from the rest of the source code.

In this work we will analyze Android SCM logs to extract and show information related to the internationalization and localization of this mobile operating system. We will analyze the changes to the source code of Android to see how l10n is being carried: who performs most of the localization work, how many people contribute to localization and if they are different people from the main developers or contributors. We will compare some metrics on the whole project, the internationalization files, and the localization files, to see if the behavior of the translation teams and processes are similar (or not) to the general software development team and processes.

The structure of this paper is as follows: next, we briefly describe how l10n is done in Android. Then, we present the methodology we have used in our study. Results are then shown in section IV and discussed in section V, including threats to validity and reproducibility issues. Following, some related efforts are presented. Finally, conclusions are drawn.

II. Localization in Android

Following the best practices for localizing Android applications described in the *Localization document*[1], developers have to create a set of default resources and alternatives that will depend on the locale. When the user runs the application, the Android system will select which resources to load, based on the device's locale at runtime.

Android developers are encouraged to introduce the text strings in English in a structured format in a file called strings.xml and located in the /res/values/ directory (that every Android module or project should have). Translators can take the text strings and localize them to their locale. As a result, another strings.xml file is generated that will be stored in the /res/values-XX directory, where XX stands for the ISO_639-1 code representation of names of languages (*fr* for French, *de* for German, etc.). The *Hello, L10N* tutorial[2] provides an example of how to build a simple localized application that uses locale-specific resources.

III. Data and Methodology

The data used for our analysis is the one provided by the MSR 2012 Challenge organizers, that includes changes in the source code management system (the git log) and a database dump of the issue tracker system. As the data is offered in XML format, we had to implement some scripts to transform it to SQL, so that they could be stored, massaged and queried using a MySQL database. The analysis is performed by a set of SQL queries. The R statistical package has been used in our analysis as well.

As Android uses git, we have the possibility to differentiate between committers and original authors; this is specially important in the case of l10n, as not all translators have write access (many of them may not know how git works).

Our methodology is based on the fact that the filenames to i18n (res/values/strings.xml) and to l10n (res/values-xx/strings.xml) files are known, so we can filter out commits to the source code management system that affect those files and analyze them and the developers that handle them. It should be noted that in addition to language l10n, there are other types of l10n such as images or screen resolution (for instance, for different types of devices) which have to be filtered. The section about “Providing Alternative Resources”[3] in the Android Developers Guide describes the different configuration qualifier names and codes, and the precedence order. The regular expression that matches language localization files (in MySQL syntax) is the following:

```
file_name like '%res/values-mcc%-__/strings.xml'
or file_name
      like '%res/values-mcc%-__-%/strings.xml'
or file_name like '%res/values-__/strings.xml'
or file_name like '%res/values-__-%/strings.xml'
```

The Linux kernel development, base of Android and also using git for managing the source code, may bias the perception of the retrieved information, since the data about its development is “merged” with the data of Android. In addition to this, the localization of the kernel does not follow the Android methodology of using resources and `string.xml` files. For this reason, the internationalization and localization data will be

[1] http://developer.android.com/guide/topics/resources/localization.html
[2] http://developer.android.com/resources/tutorials/localization/index.html
[3] http://developer.android.com/guide/topics/resources/providing-resources.html

Table I
General statistics about i18n and l10n in Android (Projs stands for Projects and Ctters stands for Committers)

	Projs	Files	Commits	Authors	Ctters
Total	275	567,357	1,771,660	12,688	1,658
Not-merge	275	-	1,603,229	12,628	1,636
Not-kernel	266	508,273	65,241	1,984	1,103
i18n	21	168	1,881	219	165
l10n	16	927	2,405	62	57
Rest	242	507,178	63,200	1,980	1,632

Table II
Effort statistics of Android developers: total developers, not-kernel developers, i18n team and l10n teams

Effort estimation	Total	Not-kernel	i18n	l10n
Files / author	44.72	256.19	0.77	14.95
Files / committer	342.19	460.81	1.02	16.26
Commits / author	139.63	32.88	8.59	38.79
Commits / committer	1068.55	59.15	11.40	42.19
Authors / committer	7.65	1.80	1.33	1.09

compared with both the total data and the "not kernel data" (total data discarding the Android kernel subprojects).

IV. Results

In total, Android is localized to 40 languages, but not all of them to the same extent. Two languages are localized in less than 10 modules, 20 languages have 11 to 20 modules translated, 10 languages have 21 to 30 modules translated and 8 languages have over 30 modules translated. The most localized language is Spanish, with 57 modules.

In Table I we can find some general statistics about the Android project. In the git log data source, each commit is related to a specific project or repository, affects a certain number of files (or zero files if it is a "merge commit"), and it is performed by a committer, in some cases different than the original author.

We present total numbers, numbers about commits affecting files (not "merge commits"), and numbers affecting files not belonging to the kernel repositories. Below them, you can find "i18n related" and "l10n related" numbers (commits changing "internationalization files" or "localization files"), and in the "Rest" row we count the projects, files, commits and people that are not performing actions to i18n files or l10n files (but change other files, again in subprojects different than the kernel). We can observe that while 0.2% of all files correspond to i18n and l10n files, the amount of activity on them (commits) is very high with 6.5% of all commits.

Table II gives some insight on the number of authors/committers per files and authors depending on the type of file. For i18n and l10n the number of developers changing the sames files is much higher than for source code.

It is common to find that the effort in free software projects is highly unequally distributed [1]. We have calculated the Lorentz curve and Gini coefficient [2] for the distribution of commits among author, for each group of developers (Figure 1).

We find that the distribution for all the teams (even discarding the kernel developers) is very unequal with Gini values above 0.8. In the case of the i18n team we find a more balanced distribution, and the localization team gives the most unequal distribution. We have to take into account that both i18n and l10n are small teams. We could expect that localization teams were where we find a more balanced distribution of work, because of the "natural" distribution of the different languages, but this unequal distribution suggests that some authors are contributing to several languages and concentrating the translation work. This makes us think that in Android we have maybe professional translators, being at the time the only translators with permissions to commit.

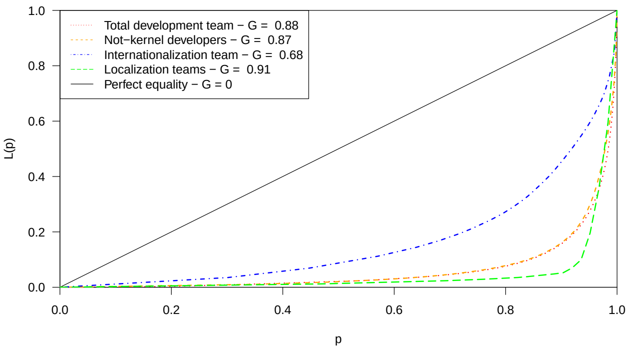

Figure 1. Lorenz Curves in Android project (authors)

For the sake of brevity, we are not including the Lorenz curve for committers, but we can say that the distribution of effort of non-kernel committers is more balanced, and the distribution of effort in i18n and l10n teams is similar to the authors teams.

But who are the translators? Do they work only on translation files or do we find that there are no "l10n teams", but general developers who perform the localization as additional contributions to the rest of the development? If we build a top-10 ranking of developers by different type of contributions, we obtain Table III for the most active developers on the global project, Table V for the most active developers acting on i18n files, and Table VI for the most active developers on l10n files[4].

Table III
Android developers: Top-ten authors by number of commits

Name	Commits
Ingo Molnar	17,272
Takashi Iwai	16,557
Bartlomiej Zolnierkiewicz	15,171
Paul Mundt	13,882
Ralf Baechle	12,596
David S. Miller	11,590
Greg Kroah-Hartman	11,388
Thomas Gleixner	11,200
Patrick McHardy	9,509
Al Viro	9,398

All of the top ten contributors to Android also belong to the top-20 most active contributors to the Linux Kernel, according to December 2010 report from the Linux Foundation[5]. In Table IV we show the top-ten authors per commits ranking, not considering this time kernel related modules[6].

If we look at the Table IV we find that some of the top modifiers of internationalization files are part of the top contributors in Android. This is what we expected, since i18n are also the files containing the strings presented to the user (prominent part of the interface of any system).

If we look at the l10n authors (see Table VI), we find that many files correspond to the initial upload of Android files to the repository. And again some of the top contributors to this kind of files are also present in the global ranking (see Table IV).

The data about authors emails are significant to know an author's affiliation to a company. For the sake of brevity, we have eliminated the

[4] For this analysis, to avoid the gate-keeper effect, as not all the contributors have access to the repositories [3], we have looked at authors, not committers

[5] https://www.linuxfoundation.org/sites/main/files/lf_linux_kernel_development_2010.pdf

[6] The author called "The Android Open Source Project" refers to the initial upload of files to the Android repositories.

Table IV
ANDROID SPECIFIC TEAMS (NOT KERNEL): TOP-TEN AUTHORS BY NUMBER OF COMMITS

Name	Commits
Marcel Holtmann	3,469
Shawn O. Pearce	3,055
Johan Hedberg	1,654
The Android Open Source Project	1,618
Xavier Ducrohet	1,186
Eric Fischer	1,156
Jean-Baptiste Queru	1,107
Matthias Clasen	1,020
Dianne Hackborn	945
Elliott Hughes	914

Table V
ANDROID I18N TEAM: TOP-TEN AUTHORS BY NUMBER OF COMMITS

Name	Commits
The Android Open Source Project	277
Eric Fischer	86
Dianne Hackborn	84
Hung-ying Tyan	59
Roy West	56
Jean-Baptiste Queru	42
Amith Yamasani	38
Bjorn Bringert	34
Dmitri Plotnikov	34
Owen Lin	31

Table VI
ANDROID L10N TEAMS: TOP-TEN AUTHORS BY NUMBER COMMITS

Name	Commits
Eric Fischer	1039
Eric Fischer (blank e-mail)	576
The Android Open Source Project	455
Kenny Root	239
Dianne Hackborn	84
Eric Fischer (nobody@android.com)	71
Hung-ying Tyan	59
Jean-Baptiste Queru	57
Roy West	56
Amith Yamasani	40

author's emails from the tables, but we discovered that all the top-ten authors modifying i18n or l10n files have a Google Inc. email address. In fact, they are almost the same group of people that perform i18n and l10n: Bjorn Bringert, Dmitri Plotnikov and Owen Lin are the number 11, 12 and 13 in the list of most active contributors in l10n files. Kenny Root, present in localization ranking, but not in our internationalization ranking, is in the 31st position in the i18n list of contributors.

V. DISCUSSION

In this paper we have observed that the activity on i18n and l10n files is different. But, contrary to our expectations, it seems that some of the key contributors to Android l10n are also key contributors to the rest of the project. From our point of view, the Android project lacks a specialized group dedicated only to l10n tasks, and benefiting from volunteer work as other free software projects have (see for instance [3]). Those contributing to l10n have to follow the same procedure as when contributing with code, making it difficult for non-technical people to join the effort of translating Android into other languages. Other free software projects use specific web platforms to ease this task and lower the burden to participate.

A. Threats to Validity

We have considered that all the folders with name matching the regular string explained in section III correspond to language localization files. However, there are some exceptions: when resources depending on Android version are required, the used codes are −vn, being 'n' a one-digit number corresponding the Android version number. This version code has precedence so it matches our discrimination condition.

Using a specific query to the data used in this study, we found that the number of files, commits and authors in this case of "versioning localization" is very low, not biasing the total numbers, but if the source data changes and metrics about "versioning localization" turn out to be significant, a new "condition string" for language localization should be designed.

B. Reproducibility of the Study

According to the reproducibility classification criteria proposed in [4], the attributes of this study are given in Table VII. Detailed information can be obtained at http://gsyc.urjc.es/∼grex/msr2012challenge.

Table VII
REPRODUCIBILITY ASSESSMENT OF THIS STUDY

Element	Assessment	Condensed Assessment
Data source	usable	U
Retrieval methodology	not usable	N
Raw dataset	usable	U
Extraction methodology	usable likely available in future flexible	U + *
Study parameters	Usable	U
Processed dataset	Usable	U
Analysis methodology	Usable likely available in future flexible	U + *
Results dataset	Usable flexible	U *

VI. RELATED WORK

Despite of the importance of i18n and l10n for the general use of end-user software, little research has been done on this topic. From the software engineering perspective, Robles et al. analyze KDE desktop environment looking for patterns in different kind of contributions by the type of file (localization files, multimedia, documentation, source code, and others) [3]. From the field of economics, Giuri et al. analyze the division of labor in free software projects and how it affects project survival and performance [5].

VII. CONCLUSION AND FURTHER WORK

In this paper we have mined the Android SCM for i18n and l10n, especially focusing to activity on i18n and l10n files and to author specialization. We have seen that i18n and l10n files show a different behavior than source code files and that in Android there is no specialized l10n team as in other projects.

It would be interesting to perform an analysis which filters the commits by subject (not by the files affected), and searches for the names of the different languages to which Android is translated, in order to compare its results with the ones obtained here. In that analysis, we could use the issue tracking system data (mining it by using the subject of the issue too). Another approach could be to try to match the SCM authors and committers with the issue reporters and fixers; but it is not clear if this is possible, since an e-mail account is required for registration in the issue tracker, but this information is obfuscated in the given data set as e-mails are provided in truncated form.

Acknowledgments

The work of Gregorio Robles has been funded in part by the European Commission under project ALERT (FP7-IST-25809) and by the Spanish Government under project SobreSale (TIN2011-28110).

References

[1] K. Crowston and J. Howison, "The social structure of free and open source software development," *First Monday*, vol. 10, no. 2, February 2005, http://www.firstmonday.dk/issues/issue10_2/crowston/.

[2] C. Gini, *On the Measure of Concentration with Espacial Reference to Income and Wealth*. Cowles Commission, 1936.

[3] G. Robles, J. M. González-Barahona, and J. J. M. Guervós, "Beyond source code: The importance of other artifacts in software development (a case study)," *Journal of Systems and Software*, vol. 79, no. 9, pp. 1233–1248, 2006.

[4] J. M. González-Barahona and G. Robles, "On the reproducibility of empirical software engineering studies based on data retrieved from development repositories," *Empirical Software Engineering*, vol. 17, no. 1-2, pp. 75–89, 2012.

[5] P. Giuri, M. Ploner, F. Rullani, and S. Torrisi, "Skills, division of labor and performance in collective inventions: Evidence from open source software," *International Journal of Industrial Organization*, vol. 28, no. 1, pp. 54–68, January 2010.