Democratizing Access to Information: An Open and Inclusive Localization Model

Amel Fraisse

Proceedings of the Language Technologies for All (LT4All) , pages 384–387
Paris, UNESCO Headquarters, 5-6 December, 2019.

Democratizing Access to Information : An Open and Inclusive Localization Model

Amel Fraisse
Univ. Lille, EA 4073 - GERiiCO
F-59000 Lille, France
amel.fraisse@univ-lille.fr

Abstract

We describe an open and inclusive localization process promoting the right of all people to use software in their mother tongue. Currently, the translation of textual resources in software is entrusted only to professional translators. This makes the localization long, expensive and intended to profitable languages. This current workflow seems impossible to apply for endangered languages for reasons of cost, and lack of professional translators. Our proposal aims at involving end users in the localization in an efficient way: while using an application, users knowing the source language (often English) could translate strings of the interface in their native language.

Keywords: Software Localization, Endangered Languages, Open and Inclusive Localization Process

Résumé

Tuvuga ubuhinga burekurira abantu bose atavangura gukoresha ubuhinga bwa none mundimi zabo kavukire. Muri kino gihe, ihindurwa ry'ibisomwa bikoreshwa murubwo buhinga bwa none rikorwa n'abahinga bo guhindura indimi babigize umwuga. Ivyo bigatuma iryo hindurwa rifata umwanya muremure, rikaba rizimvye ndetse bigatuma riba kenshi na kenshi ku ndimi zikoreshwa cane. Ico gikogwa gisa n'ikidashoboka kundimi zigeramiwe kumvo z'uburyo hamwe no kumvo zubukene bw'abahinyanyuzi b'izo ndimi. Twebwe ico dusaba n'uko abakoresha izo ndimi bohabwa urihara rukomeye muri iryo hindurwa ry'indimi, na cane cane ko ababukoresha bazi izo ndimi busanzwe buteguwemwo (kenshi na kenshi icongereza) boshobora kugerageza kubuhindura bakabushira mu ndimi z'iwabo kavukire.

1. Introduction

Currently, the translation of technical documents as well as user interface strings is entrusted only to professional translators. In practice, software publishers send original versions of the files to be localized to several professional translators. Each translator translates and sends the translated versions to the publishers. But, it seems impossible to continue in this way for most endangered languages, for reasons of cost, and quite often scarcity or even lack of professional translators (costs increase while quality and market size decrease). On the other hand, free software such as that produced by Mozilla [1] is translated by volunteer co-developers into many (more than 90) languages, in some cases more languages than commercial software. The software localization is based on the contribution of volunteers (Tong, 1987; Vo-Trung, 2004; Lafourcade, 1991). Another situation (different from the translation of technical documentation) is that of occasional volunteer translators, who contribute without an organic connection to the project. Hence, it is possible to obtain high quality translations of documents that may be over a hundred pages long (such articles of the Wikipedia encyclopedia, texts of Amnesty International and Pax Humana). Another problem of the classical localization process is that strings of the interface are often translated out of context. Hence, the choosing the appropriate translation is not always possible due to lack of context, and in such cases even a professional translator cannot produce a perfect translation. As proposed in (Boitet, 2001; Fraisse, 2010), one solution to this problem is to involve end users with a knowledge English and who, during the use of software products, translate or improve some translations proposed by machine translation (MT) systems or translation memory (TM) systems.

1.1. Current Situation

1.1.1. Crowdsourcing work force

Many online localization communities are formed and managed by volunteer localisers, software engineers, end users and in general people sharing the same motivations and aims. As a result, many projects aimed at the translation and localization of open source software and associated documents have appeared. Two quite interesting projects of this type are: the *Mozilla* localization project [2] and the Ubuntu LoCo project [3]. The *Mozilla* software set (the web browser and the email client) is available in more than 280 languages including under-resourced ones. The *Mozilla* localization model is a continuous process because each new version has new documentation and a new interface that must be translated.The Ubuntu local community team (LoCo) project is another example of a successful localization project performed by volunteer contributors. It involves 204 official local communities. Some of these communities are linked by country (Indonesia or Germany for example), others are linked by language (Catalan or Kurdish, for example), others by a geographic location (Austin, Texas USA, or Bangalore in India). Local communities involved in the localization process decide what

[1] https://l10n.mozilla.org

[2] https://www-archive.mozilla.org/projects/l10n/

[3] https://loco.ubuntu.com

needs to be localized, how and when. Translation management is entirely online through a web interface called *Launchpad Translations*. To participate in the localization of Ubuntu, volunteer contributors must be registered on the website to identify which projects are being translated, and in the context of these projects, what specific strings have to be translated. Once registered on the website the contributor can provide a translation in his native language. In fact, very often, different contributors propose several translations for the same text. But only translators ranked as *experienced* (the official members of core translators) have the right to *validate* the translations, which will then be included in *Ubuntu*.

1.1.2. Increasing demand and need for localization

Currently, in the case of commercial software, the localization decision is driven exclusively by the economic imperative. For reasons of cost, digital publishers are obliged to localize their product only to viable markets. On the other hand, the demand for localization is growing all over the world at a very fast pace, demographics and business globalization has forced the rest of world to adopt new technologies such as the Internet, wireless networks, global communications and computers. Consequently, the demand for globalization and localization is increasing. How localization will be performed and in which language is another issue. According to Sapient Globalization Report there are over 6,700 living languages in the world; the fifteen most popular languages are spoken by 49.5% of the world's population, while the other 51.5% of the world's population speak 6,600 languages. Yet, only about 6% of the world's population speak English.

2. The alternative : An open and inclusive localization model

2.1. Basic Principles

The proposed localization model is based on two basic principles:

2.1.1. Inclusive: involving volunteer translators and end users in the localization process

As we have said above, localization seems impossible for most endangered languages for reasons of cost, and quite often a scarcity or even lack of professional translators. Our solution aims at involving non-professional translators such as volunteer localisers and especially end users. These groups have the capacity to participate effectively, since they have a better knowledge of the target language (generally their native language) and of the context of use of the software. In order to motivate this type of translators and to give them a better knowledge about the use context of User Interface (UI) strings, localization should be carried out while using the software.

2.1.2. Open: from close, discontinuous, coordinated and out-of-context localization to open, continuous, uncoordinated and in context localization

Our solution aims to move from a close, discontinuous, coordinated and out-of-context localization model to open, continuous, uncoordinated and in context localization model. The basic concept consists of renouncing the idea of perfect translation and publishing rough translations with a variable quality, which will be improved incrementally during the use of the software. Therefore, the translation process will be on going and improve continuously. This solves the problem of time since users do not have to wait for the final localized version in their language. They can download, at any time, a partially localized or non-localized version of the software. Similarly, the software publisher may first publish a partially localized version that will be progressively localized through use, leading, eventually, to a complete localized version. So, the new process permits the incremental augmentation of both quality and quantity. The same principle already exists and is used by many translation communities. The best known is the Wikipedia Community, when content is added and translated continuously by contributors. So, the idea is to extend this principle to the localization filed.

2.2. Global approach

In (Fraisse, 2010), we proposed an alternative paradigm that will permit end users, volunteer translators, etc. to take part in the localization process in an efficient and dynamic way: while using the software, the end users who know the source language of the software (often but not always English) can translate or improve the previously existing translations. It is therefore an open and inclusive localization model. However, the publisher may ask professional translators and reviewers to translate the crucial parts of the software. The proposed localization model can be performed individually or collaboratively. In fact, the user has the choice to localize his/her software locally without any exchange with other users or to localize collaboratively.

2.2.1. Localizing locally

In a previous work (Fraisse et al., 2009), we have proposed a local use of our alternative localization model. That is, the model allow user to translate any string of the user interface locally. The scenario that we have envisioned is that: The user right-clicks on any string of the interface, which brings up a context menu Figure 1. That allows the user to localize the string. This is achieved by: (i) Entering a new translation, or choosing one of the proposed translations, (ii) Clicking on the *Localize* button, (iii) The interface updating in real time.

As a result, the user does not have to be connected to any resource or platform. All new translations proposed by the user are saved to a local resource file. On their next connection to the collaborative platform (we present and describe the role of the collaborative platform in the next subsection), the resource file will be analyzed and synchronized. In this way the user can submit his/her translations and get suggestions of translation proposed by other users.

2.2.2. Localizing collaboratively

End users can also localize online in order to exchange and access linguistic resources such as glossaries, dictionaries, translation memories, machine translation systems, etc. that are available on a collaborative platform. We have envisioned two possible collaboration scenarios: the first

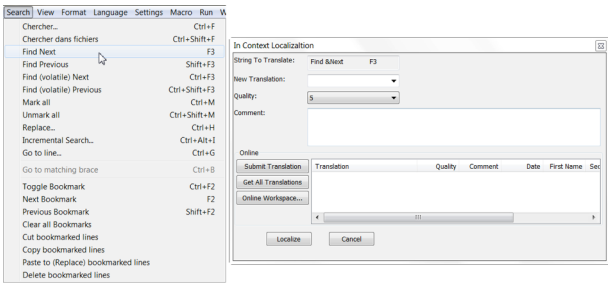

Figure 1: In context localization of the string *Find Next* of the software *Notepad++*

consists of interacting with a collaborative platform during localization; the second scenario requires localization to take place directly on the collaborative platform.

First Scenario: localization through interaction with a collaborative platform

The user right-clicks on any string of the UI to allow the string to be edited (Figure 2) : (i) The collaborative platform displays all translations proposed by other users, (ii) The user enters a new translation, or chooses one of the proposed translations, (iii) The translation is submitted to the localisers site, (iv) The user clicks on the *Localize* button, (v) The interface is updated in real time.

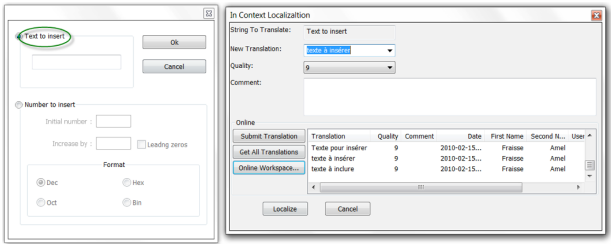

Figure 2: Collaborative and in context localization of the string *Text to insert* through interaction with a collaborative platform

Second Scenario: localizing directly on the collaborative platform

The user right-clicks on any string of the interface. This allows editing of the string directly on the collaborative platform Figure (3): (i) the user is redirected to the collaborative platform containing the string that has been chosen for translation (more details about the interface and the functionalities of the collaborative platform are described in (Huynh et al., 2008)). (ii) the user enters a new translation, or chooses one of the proposed translations. (iii) he/she returns to the original application. (iv) the interface is updated in real time.

2.3. Technical solution for an open and inclusive localization model

To enable open and inclusive localization for existing software, it is necessary to perform an internal intervention on the source code. To be as generic as possible and modify

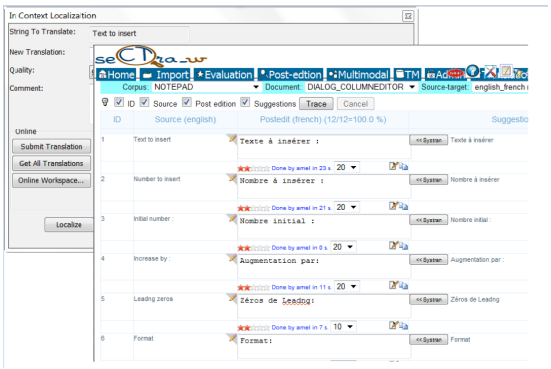

Figure 3: Collaborative and in context localization of the string *Text to insert* through interaction with the *Sectra_w* collaborative platform ((Huynh et al., 2008)).

the source code as little as possible, our modifications are only carried out on the base classes that generate all graphical user interfaces (GUIs) of the application. These modifications consist of adding new behavior adapted to the in context localization process to strings of the UI: through a simple right-click on a string in the interface, the user can choose from a list of possible translations and can add a new one, which is then updated in real time. We have implemented an open localization module pluggable into the architecture of any new or existing software developed using an object oriented programming language. The application interacts with the open localization module during the editing of user interface strings and during the update of the user interfaces. More details about the implementation of this module can be founded in our previous works (Fraisse et al., 2009, Fraisse, 2010).

3. Experiments and Results

The proposed localization model was experimented to localize the free software *Notepad++* initially localized into only 55 languages. We have decided to localize it into Vietnamese (not included in the 55 languages) and so, have created the source and the target corpus in order to import them into the *Sectra_w* collaborative platform (Huynh et al., 2008). The source corpus is based on the resource files of *Notepad++*. In the case of *Notepad++*, all strings of the user interface are stored in a single resource file named *NativaLang.xml*. We extracted and stored the UI strings in a textual file of the source corpus. To build an initial target corpus, we used *Google-Translate* to translate the UI source strings. *Notepad++* has 600 user interface strings. Three Vietnamese native speakers and users of *Notepad++* were selected to participate in the experiment. An open localizable version of *Notepad++* was installed on each of their computers and an account for each user was created on the *SECTra_w* collaborative platform (Huynh et al., 2008). For each participant the progress of his work was observed and they were asked to report the amount of time that they spent on the localization of the application. The results of this experiment are shown in Table 1.

In total the three users spent about eight hours to translate the 600 strings of the UI. We asked a professional trans-

lator about the time required to translate 600 strings form English to Vietnamese and the answer was about six hours. To assess quality of the translations, we sent the *NativeLang* file (containing the all source strings of the UI) to a professional translator and then we have asked the three user-translators to compare their translations with those performed by the professional translator. The conclusion of this comparison allowed us to deduce that the professional translator had translated approximately 36% of UI strings poorly. We attribute this to the fact that the UI strings were translated out of context.

	Day 1		Day 2	
	Time	Nb. str.	Time	Nb. str.
user 1	40mn	80	140mn	120
user 2	60mn	115	160mn	126
user 3	60mn	100	30mn	59
Total	160mn	295	330mn	305

Table 1: Duration and number of translated strings by user

4. Conclusion

We have proposed an open and inclusive localization model for new and most existing software. This new process offers a new solution to allow software localization into endangered languages and thus promoting the right of all people to use software in their mother tongue. The proposed model is an alternative to the current localization process that seems impossible to apply for most endangered languages for reasons of cost, and quite often a scarcity or even lack of professional translators. It includes volunteer localisers and specially end users in the localization process in an efficient and dynamic way: while using a software (in context), users, knowing the current language of the user interface strings, can right-click on strings to translate or improve translations. This model was experimented to localize into Vietnamese the *Notepad++* software. The localization experiment was successfully accomplished. In total the 600 strings of the user interface were localized by 3 volunteer Vietnamese native speakers.

5. Bibliographical References

Boitet, C. (2001). Four technical and organizational keys for handling more languages and improving quality (on demand). In *Proceedings of MTS2001, IAMT*, Santiago de Compostela, September.

Fraisse, A., Boitet, C., Blanchon, H., and Bellynck, V. (2009). A solution for in context and collaborative localization of most commercial and free software. In *Proceedings of LTC 2009 the 4th Language and Technology Conference, vol. 1/1*, pages 536–540, Poznań, Poland, November.

Fraisse, A. (2010). *Localisation interne et en contexte des logiciels commerciaux et libres*. Thèse de doctorat en informatique, Université de Grenoble.

Huynh, C., Boitet, C., and Blanchon, H. (2008). Sectra_w.1: an online collaborative system for evaluating, post- editing and presenting mt translation corpora. In *In proceedings of 6th Language Resources and Evaluation Conference*, Marrakech, Morocco, May.

Lafourcade, M. (1991). Odile-2, un outil pour traducteurs occasionnels sur macintosh. In *Presses de l'université de Québec*, Université de Montréal.

Tong, L. (1987). The engineering of a translator workstation. *Computers and Translation*, pages 263–273.

Vo-Trung, H. (2004). *Méthodes et outils pour utilisateurs, développeurs et traducteurs de logiciels en contexte multilingue*. Thèse de doctorat en informatique, Institut National Polytechnique de Grenoble.