A Comparative Study of Two FOSS Localisation Tools: Pootle and Virtaal

SIANO, Valeria

Abstract

This thesis is focused on the analysis and comparison of two popular FOSS localisation tools: Pootle, a web-based system, and Virtaal, a stand-alone tool. As FOSS programs are not always localised by professionals, the main purpose of our study was to determine which tool would be better for a novice translator who wants to start collaborating on a project. By following the EAGLES final report, we evaluated three main characteristics (functionality, efficiency, usability) together with their sub-characteristics (suitability, accuracy, time behaviour, understandability, learnability, operability, attractiveness) by the means of a feature inspection, a scenario test and a task-specific questionnaire. Since we would also like to have the point of view of active users of the two tools, we subsequently performed an online survey. Results showed that Virtaal and Pootle scores were similar in both our study and online survey. Consequently, we can argue that both tools are appropriate for beginners.

Reference


Available at:
http://archive-ouverte.unige.ch/unige:75916

Disclaimer: layout of this document may differ from the published version.
Valeria Siano

MA Thesis

A Comparative Study of Two FOSS Localisation Tools: Pootle and Virtaal

Supervisor: Dr. Lucía Morado Vázquez
Examiner: Marianne Starlander

Faculty of Translation and Interpreting, University of Geneva
Department of Translation Technology

August 2015
J'affirme avoir pris connaissance des documents d'information et de prévention du plagiat émis par l'Université de Genève et la Faculté de traduction et d'interprétation (notamment la Directive en matière de plagiat des étudiant-e-s, le Règlement d'études de la Faculté de traduction et d'interprétation ainsi que l'Aide-mémoire à l'intention des étudiants préparant un mémoire de Ma en traduction).

J'atteste que ce travail est le fruit d'un travail personnel et a été rédigé de manière autonome.

Je déclare que toutes les sources d'information utilisées sont citées de manière complète et précise, y compris les sources sur Internet.

Je suis conscient-e que le fait de ne pas citer une source ou de ne pas la citer correctement est constitutif de plagiat et que le plagiat est considéré comme une faute grave au sein de l'Université, passible de sanctions.

Au vu de ce qui précède, je déclare sur l'honneur que le présent travail est original.

Nom et prénom : SIANO Valeria

Lieu / date / signature : Genève, le 15 Juillet 2015
L’importante non è il voto, ma rispettare il proprio dovere e portarlo avanti fino in fondo.

P. Borsellino
Acknowledgements

Me gustaría agradecer a la Dra. Lucía Morado Vázquez por todos los consejos que me ha dado, por lo servicial y, sobre todo, paciente, que ha sido conmigo y por haberme transmitido su pasión por la localización.

I would like to thank Marianne Starlander, who was pleased to accept being on my jury.

J’aimerai remercier les administrateurs de Sympa pour avoir héberger mes projets sur leur serveur Pootle. En particulier, je souhaiterais remercier David Verdin, car sans son aide et sa disponibilité mon étude n’aurait pas pu être conduite.

I would also like to thank participants to this evaluation, who decided to spend some of their time helping me.

I would like to thank the staff at Translated for authorising me to use an image taken from their website.

Vorrei ringraziare soprattutto la mia famiglia, che mi è stata vicina durante questi lunghi anni di studio, nonostante la lontananza, che mi ha sostenuto nei momenti di sconforto e che mi ha insegnato che nella vita bisogna lottare per ciò che si vuole, senza mai perdere di vista i propri valori.

Vorrei ringraziare i miei amici ( sperando lo siano ancora dopo i miei mesi di “ritiro ascetico” per via della tesi), che non hanno mai smesso di incoraggiarmi.

Ed infine vorrei ringraziare te, che sai già tutto e che hai creduto in me fino alla fine.
# Table of Contents

General Introduction ................................................................. 9  

1. Internationalisation and Localisation ........................................ 12  
   1.1. Introduction ......................................................................... 12  
   1.2. Defining the Terms .............................................................. 12  
   1.3. The Origins of Localisation .................................................. 16  
   1.4. Software Localisation ......................................................... 17  
   1.5. Conclusion ........................................................................... 23  

2. Free and Open Source Software ............................................... 24  
   2.1. Introduction ......................................................................... 24  
   2.2. A Problematic Definition of Free Software ............................ 24  
   2.3. Free Software vs. Proprietary Software ................................ 27  
   2.4. Free Software vs. Open Source ........................................... 30  
   2.5. Conclusion ........................................................................... 31  

3. FOSS Localisation ...................................................................... 32  
   3.1. Introduction ......................................................................... 32  
   3.2. Localisation Workflow ......................................................... 32  
   3.3. Tools .................................................................................. 35  
   3.4. Aspects and Motivations ...................................................... 37  
   3.5. Success Stories .................................................................... 41  
   3.6. Conclusion ........................................................................... 42  

4. Evaluating and Comparing ....................................................... 43  
   4.1. Introduction ......................................................................... 43  
   4.2. Past Evaluation Methods ..................................................... 43
4.3. Methodology ................................................................. 45
4.4. Criteria ............................................................................. 46
4.5. Tests ............................................................................... 49
4.6. Conclusion ....................................................................... 50
5. Evaluation Design and Requirements ....................... 51
  5.1. Introduction ...................................................................... 51
  5.2. Purpose ........................................................................... 51
  5.3. Participants ...................................................................... 52
  5.4. Materials .......................................................................... 52
    5.4.1. NoteFly ....................................................................... 55
    5.4.2. BB FlashBack Express ............................................. 57
  5.5. Localisation Tools .......................................................... 57
    5.5.1. Virtaal ......................................................................... 57
    5.5.2. Pootle ......................................................................... 62
    5.5.3. Summary and Comparison ........................................ 64
  5.6. Conclusion ....................................................................... 68
6. Evaluation process .............................................................. 69
  6.1. Introduction ...................................................................... 69
  6.2. Tests with Virtaal ............................................................. 69
    6.2.1. Pilot Study .................................................................... 70
    6.2.2. Evaluation Implementation ............................... 71
  6.3. Tests with Pootle ............................................................... 73
    6.3.1. Pilot Study .................................................................... 74
    6.3.2. Evaluation Implementation ............................... 76
  6.4. Conclusion ....................................................................... 78
7. Results ............................................................................... 79
  7.1. Introduction ...................................................................... 79
  7.2. Objective Approach ....................................................... 79
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.2.1.</td>
<td>Time Behaviour</td>
<td>79</td>
</tr>
<tr>
<td>7.2.2.</td>
<td>Operability</td>
<td>81</td>
</tr>
<tr>
<td>7.3.</td>
<td>Subjective Approach</td>
<td>86</td>
</tr>
<tr>
<td>7.3.1.</td>
<td>Suitability</td>
<td>86</td>
</tr>
<tr>
<td>7.3.2.</td>
<td>Accuracy</td>
<td>89</td>
</tr>
<tr>
<td>7.3.3.</td>
<td>Efficiency</td>
<td>90</td>
</tr>
<tr>
<td>7.3.4.</td>
<td>Understandability</td>
<td>91</td>
</tr>
<tr>
<td>7.3.5.</td>
<td>Learnability</td>
<td>91</td>
</tr>
<tr>
<td>7.3.6.</td>
<td>Attractiveness</td>
<td>92</td>
</tr>
<tr>
<td>7.4.</td>
<td>Summary</td>
<td>95</td>
</tr>
<tr>
<td>7.5.</td>
<td>Findings</td>
<td>96</td>
</tr>
<tr>
<td>7.6.</td>
<td>Conclusion</td>
<td>97</td>
</tr>
<tr>
<td>8.</td>
<td>Online Survey</td>
<td>98</td>
</tr>
<tr>
<td>8.1.</td>
<td>Introduction</td>
<td>98</td>
</tr>
<tr>
<td>8.2.</td>
<td>Purpose</td>
<td>98</td>
</tr>
<tr>
<td>8.3.</td>
<td>Users’ Profile</td>
<td>98</td>
</tr>
<tr>
<td>8.4.</td>
<td>Lime Survey</td>
<td>101</td>
</tr>
<tr>
<td>8.5.</td>
<td>Online Survey Analysis</td>
<td>103</td>
</tr>
<tr>
<td>8.5.1.</td>
<td>Suitability</td>
<td>104</td>
</tr>
<tr>
<td>8.5.2.</td>
<td>Accuracy</td>
<td>106</td>
</tr>
<tr>
<td>8.5.3.</td>
<td>Efficiency</td>
<td>107</td>
</tr>
<tr>
<td>8.5.4.</td>
<td>Understandability</td>
<td>107</td>
</tr>
<tr>
<td>8.5.5.</td>
<td>Learnability</td>
<td>108</td>
</tr>
<tr>
<td>8.5.6.</td>
<td>Attractiveness</td>
<td>108</td>
</tr>
<tr>
<td>8.6.</td>
<td>Results comparison</td>
<td>112</td>
</tr>
<tr>
<td>8.7.</td>
<td>Conclusion</td>
<td>113</td>
</tr>
<tr>
<td>9.</td>
<td>Conclusions and Future Work</td>
<td>114</td>
</tr>
<tr>
<td>Bibliography</td>
<td>117</td>
<td></td>
</tr>
<tr>
<td>Annex I – Information Sheet, Consent Form</td>
<td>125</td>
<td></td>
</tr>
<tr>
<td>Annex II – General Questionnaire</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Annex III – Virtaal Instructions</td>
<td>133</td>
<td></td>
</tr>
</tbody>
</table>
Annex IV – Pootle Instructions.................................139
Annex V – Virtaal Final Questionnaire...........................145
Annex VI – Pootle Final Questionnaire............................154
General Introduction

‘Those of us who understand the language of the technology are able to use it, while other people might be entirely locked out, or unable to fully enjoy the benefits that the technology brings.’

(Wolff Friedel, Effective change through localisation)

The Internet has changed many aspects of our daily life developing as an important cultural and social phenomenon. For this reason, its effects on society are often compared to those of the printing press, which is somehow considered the forerunner of computer networks and technological progress in general (Hauben, 1997).

Language industry developed in the 1990s when Internet expansion, electronic information exchange and liberalisation of the telecommunication market gave rise to new information and communication technologies. The so-called Information Age has been in fact an authentic revolution in the way of communicating, especially for people dealing with different languages like translators and interpreters. Information management is the heart of any economic activity and it facilitates opportunities for interlingual communication, thus increasing language contacts. Although multilingualism becomes an everyday experience for millions of people, language is definitely one of the few remaining barriers in the Information Age. With the ISO 9000\(^1\), business services and processes have been standardised around the world, providing customers with a high level of quality. In this way, quality standards have increased, international trade has grown and companies’ efforts to overcome cultural barriers have become crucial. Now more than ever, the success of a product in foreign markets depends on companies respect for the customers’ culture. Proprietary software internationalisation and localisation have thus become an important issue, leading to the creation of in-house localisation teams by large multinational

---

\(^{1}\) ‘The ISO 9000 family addresses various aspects of quality management and contains some of ISO’s best known standards. The standards provide guidance and tools for companies and organizations who want to ensure that their products and services consistently meet customer’s requirements, and that quality is consistently improved.’

companies and the development of localisation service providers (LSPs), mostly in Ireland (Esselink, 2000), for companies that want to outsource their projects. At the opposite side, with the creation of Stallman’s GNU Project and the development of Free and Open Source Software (FOSS) movement, users became more and more engaged in remote, collaborative work, including FOSS programs development, internationalisation and localisation.

This thesis is focused on FOSS localisation, and, in particular, on the analysis and comparison of two popular programs in FOSS localisation projects which were developed by the same organisation: Pootle2, a web-based system, and Virtaal3, a stand-alone tool. As FOSS programs are not always localised by professionals, the main purpose of this study was to figure out which system would be better for a non-professional translator who wants to start collaborating on a project. Since we would also like to have a comparison from the point of view of active users of the two tools, we performed an online survey.

The structure of this thesis is organised in nine chapters. The first chapter provides an introduction and general overview of software localisation. The second chapter is based on Free and Open Source Software: it discusses the controversial definition of Free Software by analysing its differences from proprietary software and Open Source Software. In the third chapter, we focus on FOSS Localisation by describing workflows, tools and formats used in FOSS projects. At this stage we also provide some examples of FOSS success stories. In the fourth chapter, we describe the approach taken for the comparative study of Virtaal and Pootle, that is to say that the evaluation methodology, criteria and tests are defined. The fifth chapter is an introduction to the evaluation process: the purpose of this study is clarified and a description of users’ profile and materials is provided. Then, Virtaal and Pootle are introduced by providing an overall comparison of their features. The sixth chapter is focused on the evaluation process, where the performed experiments with Virtaal and Pootle are described. In chapter 7, we analyse and compare evaluation results by following some predefined metrics. A summary is then provided at the end of the chapter. Chapter 8 presents an online survey, which was performed in order to compare

---
2 You can access the official Pootle server at [http://pootle.locamotion.org](http://pootle.locamotion.org).
3 Virtaal can be downloaded at [http://virtaal.org](http://virtaal.org).
different points of view on Virtaal and Pootle, in relation to the level of participants’ experience with the analysed tools. Finally, a conclusion is drawn in chapter 9, where the findings and limitations of our study are described and suggestions for future work are provided.
1. Internationalisation and Localisation

1.1. Introduction

In a world where cultural and linguistic barriers are more and more overthrown, internationalisation and localisation become essential. Despite globalisation, users are, in fact, more likely to identify with a product when it is adapted to their linguistic and cultural features (Riediger and Galati, 2012: 4).

This chapter provides an overview of software localisation. It begins with a definition of the terms Internationalisation and Localisation showing the difference between the two processes (section 1.2.). In section 1.3., we will analyse the origins of localisation, describing industry development through its milestones and markets expansion. The chapter then focuses on the characteristics of software localisation, analysing its differences from translation and highlighting the most important issues and benefits (section 1.4.).

1.2. Defining the Terms

According to the Localisation Industry Standards Association (LISA)\(^4\), internationalisation, which is usually shortened to I18n – where 18 stands for the letters between I and n – is ‘the process of generalizing a product so that it can handle multiple languages and cultural conventions without the need for re-design. Internationalisation takes place at the level of program design and document development.’ In fact, this process paves the way for the eventual localisation of a product and simplifies this latter by reducing its time and costs.

Performed by software developers, I18n is also a strategy indispensable for the so-called SimShip, that is to say the simultaneous shipment of web sites, software and documentation in different languages and locales. According to Sikes (2009), the internationalisation process consists of three levels of action:

\[^4\] The Localisation Industry Standards Association (LISA) was founded in 1990 in Switzerland as a means of ‘promoting the localization and internationalization industry and providing a mechanism and services to enable companies to exchange and share information on the development of processes, tools, technologies and business models connected with localization, internationalization and related topics’ (Esselink, 2000). In 2011, LISA disappeared because of insolvency problems (Beninatto, 2011).
- The technical level involves the adaptation to global norms like, for example, the Unicode Standard, a universal character-encoding system;

- The structural level consists of the separation of user interface text from source code: translatable strings are centralised in separate resource-only .dll files to avoid code breaking during the localisation process. Therefore, simply replacing these program files with their localised version might be enough to run a program in another language;

- The cultural level is characterised by terminology, style and design standardisation. In particular, all language, cultural and local conventions, such as fixed date, time, currency or address formats and jargon or other specific languages, should be removed to ensure reusability. Developers also have to avoid specific images, icons or graphics, and text in the user interface should not be submitted to size restrictions, thus creating an essentially neutral product.

After describing internationalisation, we are now providing a definition of localisation. The term is usually abbreviated to L10n (10 stands for the letters between L and n) and derives from locale. This latter represents both a region and a language, but also various cultural assumptions of a given language and geographic area, since considerable cultural differences also exist within countries where different languages are spoken. According to the ISO 639-1\(^5\) and ISO 3166-1\(^6\) standards, a locale is indicated by a two-letter

---

\(^5\) ‘ISO 639 is the International Standard for language codes. The purpose of ISO 639 is to establish internationally recognised codes (either 2, 3, or 4 letters long) for the representation of languages or language families.’


\(^6\) ‘ISO 3166 is the International Standard for country codes and codes for their subdivisions.’ It consists of three kinds of code:

- Codes for countries can be specified with a two-letter (alpha-2), three-letter (alpha-3) or three digit numeric code (numeric-3);
- Codes for subdivisions (provinces or states) include the countries two-letter code and a string of up to three characters;
- Formerly used codes are represented by a four-letter code (alpha-4).
code in lowercase representing a language and a two-letter country identifier in uppercase. We have, for example, it/IT for *Italian/Italy* and it/CH for *Italian/Switzerland*. Localisation is the translation and cultural adaptation of digital products and accompanying materials to one or more locales and requirements of a foreign market. It provides services and technologies for the management of multilingualism across the digital global information flow (Schäler, 2009) and its main purpose is to make users interact with products that seem to be developed in and for their locale. Localisation choices concern the following aspects: language and its regional varieties, writing conventions (date and time format, capitalisation, addresses, punctuation, etc.) and cultural conventions, including symbols, logic structures, colours, images and sounds.

The Localisation Industry Standards Association (LISA) refers to localisation as the process of ‘taking a product and making it linguistically and culturally appropriate to the target locale (country/region and language) where it will be used and sold.’ Companies decide to localise their products mainly for three reasons: because of foreign consumers’ demand of localised versions of the product (the reactive approach), business growth into new foreign markets, where localisation becomes a competitive advantage (the strategic approach) (Dunne, 2011: 17), and for legal constraints, since distribution of non-localised products is not allowed in many countries. Cultural adaptation is a very important aspect of the localisation process and it is generally undertaken by marketing and commercial departments to best suit consumers’ customs and to consequently influence their buying decisions. In this sense, we are now analysing two different localisation examples. In Figure 1 we can find three covers of the video game *MotoGP 13* for *PlayStation 3*, where different riders are represented, depending on the distribution country. We notice that Valentino Rossi and Andrea Dovizioso are in the foreground of Italian version, while Carl Crutchlow is represented as the main character of English version. A part from the rider Stefan Bradl in the background of German version, we also have to draw attention to the age rating symbol by

URL: [www.iso.org/iso/country_codes.htm](http://www.iso.org/iso/country_codes.htm) (accessed 20.03.2014).
the USK\textsuperscript{7}, which is specific to games to be sold in Germany because of law restrictions.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig1.png}
\caption{Three localised versions of MotoGP 13. (From left to right: Italy, UK and Germany)}
\end{figure}

The second example (Fig. 2) consists of the Italian version of McDonald’s website\textsuperscript{8}. First of all, the insalata di pasta advertised on the homepage is a distinctive meal of Italian cooking and it definitely represents an innovation among typical McDonald’s products. Then, we can notice the partnership with Barilla, a well-known pasta producer, which is also seen by Italians as a symbol of their culture and traditions. Finally, colours are another significant point to be highlighted: azure represents, in fact, Italian National Football Team, also known as the Azzurri. Since the most popular sport in Italy is definitely football, we can notice once again a significant presence of cultural adaptation.

\textsuperscript{7} The abbreviation stands for Unterhaltungssoftware Selbstkontrolle (the Entertainment Software Self-Regulation Body, in English), a German organisation which classifies computer games so that they cannot be sold to children and young people if the contents are not safe for them. The classification is based on the Jugendschutzgesetz and age rating symbols are assigned by youth protection authorities. In this particular case, the game is classified as ‘Approved without age restriction in accordance with Art. 14 German Children and Young Persons Protection Act (JuSchG)’. This means that it is suitable for everybody, which is typical of Sports Games, except from martial arts and combat sports.

\textsuperscript{8} Accessed 05.06.2014.

URL: \url{http://www.usk.de/en/classification/age-rating-symbols} (accessed 5.06.2014)
Besides business and legal reasons, localisation is above all an important process for the preservation of linguistic and cultural diversity, providing people from anywhere in the world with access to digital contents independently of their languages and customs (Schäler, 2004: 28).

1.3. The Origins of Localisation

Personal computing and Internet expansion encouraged American software companies to look for new markets where they could distribute their products. Europe was seen as the first potential market because of its richness, and the need for design, translation and adaptation of products to language and cultural conventions in different locales became a real issue for companies.

Therefore, existing companies were provided with localisation departments, while new ones, such as INK (now Lionbridge) or IDOC (now Bowne), were created to deal with software localisation and technical documentation translation. The mid-1980s marked the origins of two different specialised firms: Single Language Vendors (SLVs), working in one target language with often limited technical skills, and Multi Language Vendors (MLVs), experts in technology and various target languages (Esselink, 2000). In the 1990s,
a GILT (Globalisation, Internationalisation, Localisation, Translation) sector developed: globalisation needed internationalisation to facilitate products localisation and translation into other languages for possible distribution on markets they were not created for. Over the past 20 years, localisation has been essential for the development of the multilingual information society, thus becoming the ‘first industrial sector where language resources have been used widely and consistently on large-scale commercial projects’ (Schäler, 2003: 5).

Ireland is the centre and leader of localisation industry since the late 1990s, when government grants, competitive labour costs and skilled, motivated people, issued by several investments in education, favoured this development (Esselink, 2000). In the country, we can also find most of the main software companies, such as Microsoft, Oracle and SDL, as well as the Localisation Research Centre (LRC) at the University of Limerick, created in 1997 to connect the academic world with digital publishers and localisation service providers (Schäler, 2014).

1.4. Software Localisation

Digital products are mostly localised from English, since the majority of them is developed in the USA, but even in other countries English version is generally used as a source for localisation into other languages (Esselink, 2000). Developers and users from across the world also prefer using this common language for communication because it is more practical (Wolff, 2011), but a lot of people are not fully proficient in English, do not speak it at all, or simply prefer working with products in their own language. There are different localisable digital products, including software, video games and websites. Before describing in details the specific case of Free and Open Source Software, which is the focus of this thesis, we should briefly explain the concept of software localisation and point out its main features.

The translation of all graphical user interface (GUI) components of a program is the heart of software localisation: dialog boxes, menus, strings containing error or status messages, questions and tooltips are in fact translated and reviewed before anything else, including online help or documentation. According to Esselink (2000), there are two ways of localising a software program:

1. Text-only resource files localisation by means of a text editor, a translation memory tool, or a resource editor. Localised files are subsequently compiled into binary
program files;

2. Program files – also called satellite .dll or resource-only .dll files – localisation through a resource editor or a software localisation tool. In this way, it is possible to prevent code breaking in the text-only resource files and since translation is instantly displayed, dialog boxes can be directly resized when necessary.

As observed before, localisation is more than simple translation. Translators and revisers work above all with texts, like books or documents, while L10n concerns different sources, only partially composed of text, thus involving other tasks, such as project management, software engineering, testing, and desktop publishing (Esselink, 2000). Given the considerations above, it is important to underline that, in general, books or texts always appear the same way to target readers, as they have a determined structure before and after translation, on the contrary software localisation is characterised by unpredictability, because the running program introduces some changes in the text and users might experience target version differently, since they might not interact with it the same way nor in a fixed order. Therefore, software localisation needs specialists, like engineers and localisers, CAT and localisation tools to help them with their projects by automating some processes, and standards such as TMX, TBX or XLIFF to facilitate exchanges without data corruption or loss (FLOSS MANUALS, 2011). Despite the many differences, translation and software localisation have the same language issues: both translators and localisers must pay attention to orthography, punctuation, register and grammatical style, while remaining faithful to the source text and the target language (Wolff, 2011). Some specific guidelines are needed to prevent incoherence, which would otherwise badly influence productivity and quality. Since software localisation concerns more technical texts, terminology is also very important to ensure coherence and transparency in users’ experience (ibid).

Although translation of user interfaces is an essential part of software localisation, adaptation to local contexts implies many other changes, which are at the basis of the most common localisation issues, including number and date formats, currency and units of measurement, images and colours, sorting rules, lists and hot keys – also called mnemonic, accelerator or access keys —, that is to say a combination of commands to quickly access menu or dialog box options (Esselink, 2000). First of all, localisers have to identify translatable
and untranslatable elements. In general, text between double quotes in software resource files needs to be translated, but there may be some exceptions, like for example “MS Sans Serif”, which is the name of the used font, or “Button”, which must not be changed as it refers to the control type (Esselink, 2000: 61). Aiming to provide localisers with useful information, software engineers usually insert some comment lines, which are translated or left unchanged depending on given instructions. The main elements which should not be translated can be found in the following list by Wolff (2011: 35-43):

- **Email addresses** (apart from those ones used as examples, like user@...) and **URLs** (only language indicators should be replaced);

- **Abbreviations** concerning file format or some other technical names, like for example TBX;

- **Trade marks** should be left unchanged, but products or companies names can be translated, if they are words with a specific meaning or an equivalent in the target country;

- **Function calls** or other aspects of a network protocol, programming or query language (mostly written in capital letters or between quotation marks);

- **Command-line options**, except for its arguments, which can be translated, because they are just place-holders;

- **Escape characters**, also called control or formatting codes. The table below shows different types of escape characters and provides information on the consequent formatting changes in the compiled program.

<table>
<thead>
<tr>
<th>Escape sequence</th>
<th>Effect</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>Newline</td>
<td>A new line in the text (similar to pressing Enter). It might be shown with the ¶ in your translation program.</td>
</tr>
<tr>
<td>\t</td>
<td>Tab</td>
<td>Add a tab marker at that point in the text. This is sometimes used to align text in columns.</td>
</tr>
<tr>
<td>\r</td>
<td>carriage</td>
<td>This is usually used together with \n in software for</td>
</tr>
</tbody>
</table>
If you need a real backslash then you need to escape the backslash. The computer will only print one backslash.

In some cases a double quote has a special meaning, and should be escaped if you really want a double quote.

Special characters like © or ™. Programmers might use \u to refer to the special character. Here xxxx is the hexadecimal or decimal value for the specific Unicode character. You probably want to leave it as is.

In some XML documents, you might need to represent the apostrophe and some other characters with XML entities. For example:

<a href="index.html" title="Rock &apos;n Roll">link</a>

Table 1: Types of escape character (Wolff, 2011: 37).

For what concerns markup languages, in particular HTML and XML, translation is characterised by a high risk of code corruption, since translatable text cannot be easily recognised. It is very important to leave all tags, attributes and most of values untranslated. Variables, that is to say ‘placeholders for numbers or some text that will be filled into a message when the program is running’ (Wolff, 2011: 41), are another important element to leave unchanged.

In Table 2, we can find some common variables, such as %s, usually representing a text string, or %d, referring to a number written in digits.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>%s</td>
<td>String</td>
</tr>
<tr>
<td>%d</td>
<td>decimal integer</td>
</tr>
<tr>
<td>%ld</td>
<td>decimal long integer</td>
</tr>
<tr>
<td>%x</td>
<td>integer in hexadecimal form</td>
</tr>
<tr>
<td>%g</td>
<td>floating point value</td>
</tr>
<tr>
<td>%u</td>
<td>Unicode character</td>
</tr>
<tr>
<td>%p</td>
<td>page number</td>
</tr>
</tbody>
</table>

Table 2: List of the most common variables represented by Esselink (1998: 22).

Sometimes, localisers have to deal with three serious variables issues which concern reordering, omitting, times and dates. According to Wolff’s (2011) analysis, in the first case, translation implies a change in variables order to best suit the target language style. For
example, if the variable %s came first in the original string, now that the order has changed, it must be indicated by %1$s. Omitting helps localisers with missing expressions in some target languages and translation of some plural forms, but it is not always possible. Thus, we are forced to modify the variable in question to prevent it from displaying, that is to say that %od , for example, must be replaced by %.0d. For what concerns times and dates issues, if they involve different formats, localisers should use other variables in the target strings than those of the source text.

Besides language and technical issues, software localisation let companies overthrow language barriers, and at the same time it provides people with access to everyday activities in the digital world. In fact, as remarked by Dalvit et al. (2008), without localisation, technology, which is somehow already “foreign” to many by its own, would become incomprehensible for people who cannot speak the language in question. Software companies firstly localise their products into FIGS (French, Italian, German, Spanish) and Japanese, then into Swedish, Norwegian, Danish, Dutch or Brazilian Portuguese. Niode (2009) remarks that Central Europe, China and India have recently become ‘the central hub for the world wide localization industry’, thanks to cheaper employment for foreign multinationals. For Schäler (2010), China and India are likely to develop into major producers of digital content very soon. This is confirmed by the T-Index\(^9\) projection (Fig. 3), which shows that in 2016 China will be at the first place with a 20.1% market share, overtaking United States (15.6%). Figure 3 also shows that Brazil, which is another emerging country, would reach the fourth place of the Top 10, leaving France, United Kingdom and Germany behind. In general, we notice that the T-Index Study shows an authentic growth in all emerging countries such as China, Brazil, Poland, Russia, India, Nigeria and Vietnam.

---

\(^9\) The T-Index Study [www.translated.net/en/languages-that-matter](http://www.translated.net/en/languages-that-matter) was carried out by Translated, a language service provider founded in 1999 that manages 68.000 translators in over 110 countries for 26.000 customers. In 2008, Translated was selected by Eurispes as representative of Italy’s Excellence. Worldwide, Translated ranks amongst the most innovative LSPs in the development of productive processes to help translators in their daily endeavours. The T-Index is a percentage value, resulting from the Internet population and its estimated GDP per capita, to indicate the online sales potential in each country. It concerns websites, but we are taking them into consideration to analyse the development of localisation industry in different countries.
In 2000, the majority of software companies produced more than 60% of the overall revenues from international business departments, that is to say that they had more incomes from localised products sales than from the original product (Esselink, 2000). Therefore, language choice has always depended on the Gross National Product (GNP) of the target country rather than the number of speakers. As a result, companies translate for 2 billion people in 15 languages, including, for example, Danish, accounting for 5 million speakers, but not Amharic, Ethiopia national language spoken by 27 million people (Schäler, 2009). At the opposite side, social localisation covers the remaining 5 billion people, that is to say 70% of world’s citizens, speaking 6,985 languages (Schäler, 2013). Users are increasingly using their experience and skills to localise products for those speakers around the world who ‘have been ignored by the mainstream translation and localization industry because they do not represent a viable business case’ (O’Brien and Schäler, 2010). This ideology is shared by FOSS community which undertakes this type of localisation to support peoples, independent of their socio-economic status, language and country. In this
sense, software localisation not only provides users with access to information and digital content, but it also ensures language and cultural preservation.

1.5. Conclusion

This chapter was based on Internationalisation and Localisation. We have firstly defined and analysed the two processes to underline the differences between them (section 1.2.). Then, we have described the origins of localisation industry (section 1.3.). In section 1.4., we have focused on software localisation, in particular we have analysed its features, its differences from translation and its main language and technical issues. Since FOSS localisation is at the centre of this study, we have finally introduced the FOSS community and the concept of social localisation in opposition to software companies and the localisation industry.
2. Free and Open Source Software

‘Computers are tools for human beings. Ultimately, therefore, the challenges of designing hardware and software must come back to designing for human beings—all human beings. This path will be long, and it won’t be easy. But I think the hacker community, in alliance with its new friends in the corporate world, will prove up to the task. And, as Obi-Wan Kenobi might say, “the Source will be with us”’.

(Raymond Eric S., Revenge of the Hackers)

2.1. Introduction

Source code is at the basis of any software application, as it is composed of commands, in a particular programming language, illustrating tasks to be performed by computers. Most software companies do not provide users with the source code of their programs, but only with compiled instantiations. Those compiled and executable versions are bonded by licences that specify the terms of use. This strategy ensures them the control on the development of their software and the release of new versions. On the contrary, Free and Open Source Software programs do not involve restrictions: this kind of software can be freely copied, modified, localised and distributed by users for any purpose (Free Software Foundation, 2012). Nowadays large communities of volunteer programmers and localisers are engaged in FOSS projects.

This chapter provides an overview of Free and Open Source Software. After discussing the controversial definition of Free Software (section 2.2.), it describes proprietary software (section 2.3.) and Open Source Software (section 2.4.) by showing the differences between these three kinds of programs.

2.2. A Problematic Definition of Free Software

In order to fully understand the meaning of Free Software, first of all we should briefly outline the origins of this expression. As stated by Richard Stallman, a computer scientist
working at the MIT Artificial Intelligence Lab\(^{10}\) in 1970, ‘…sharing of software . . . is as old as computers, just as sharing of recipes is as old as cooking’ (Stallmann, 2002: 17). However, the term Free Software was coined in 1983. At that time, Stallman had some problems with a printer at the AI Lab and wanted to fix the driver software, but developers refused to provide him with a copy of the source code, so he decided to start the GNU Project\(^{11}\) to create a free Unix-like operating system that would be a valuable alternative to proprietary software (Parker, 2011). In order to promote this cause, Stallman founded the non-profit Free Software Foundation (1985) which still raises funds from donations and selling GNU CD-ROMs, T-shirts and manuals. In 1991, he decided to combine the GNU system with Linux, a Unix-compatible kernel developed by Linus Torvalds, a Finnish computer scientist, thus creating a complete free operating system called GNU/Linux (Stallmann, 2002). This achievement demonstrated that a community of volunteers was able to establish real alternatives to closed source codes by developing software applications which could compete with the most advanced proprietary software. Even if there is still little doubt concerning quality and viability of this kind of applications, ‘free software has secured a spot on the servers that control the Internet, and—as it moves into the desktop computer market—is a threat to Microsoft and other proprietary software companies’ (Stallmann, 2002: 1).

As mentioned before (2.1), the expression Free and Open Source Software refers to a program that users can freely download, run, change, copy and redistribute with or without changes, including for example localisation. The term free might be misunderstood, as people might think it concerns price and usually refer to it as a synonym of costless. On the contrary, ‘free software is a matter of liberty, freedom’ and in this sense free stands for ‘limited in its control by others’ (Stallmann, 2002: 11). According to Stallman, in order to understand this

\(^{10}\) The MIT Artificial Intelligence Lab was formed in 1959 at the Massachusetts Institute of Technology. This research laboratory was specialised in finding ‘new methods for image-guided surgery and natural-language-based Web access’ and developing ‘bacterial robots and behavior-based robots that are used for planetary exploration, military reconnaissance and in consumer devices’. In 2003, the AI Lab and the Computer Science Lab merged to form the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL).


\(^{11}\) Following a hacker tradition, the name GNU is a recursive acronym which stands for GNU’s Not Unix. (Stallmann, 2002)
concept, ‘a much more fundamental sense’ of the term must be considered: free not as in free beer, but in terms of free speech or free labor. Free and Open Source Software can be distributed for free or at some cost, because ‘since free software is not a matter of price’, selling copies is not a contradiction and ‘a low price isn’t more free, or closer to free’ (Stallmann, 2002: 65). Users’ freedom to copy, modify or share programs is at the basis of the definition, as a matter of fact copyright restrictions are the reason why some free (this time as in free beer) programs, for example freeware, are not considered Free and Open Source Software. The figure below is a conceptual map clarifying Free and Open Source Software definition, characteristics and aims.

![Conceptual Map of Free Software](image)

In particular, we can notice that, according to the Free Software Foundation (2012), a program is Free Software if users have the following essential freedoms:

- The freedom to use the program, for any purpose (freedom 0). There are no
software license terms to be accepted, as users have the right to do whatever they want with Free and Open Source Software.

- The freedom to study the program and adapt it to users’ needs, for example by fixing bugs or adding new features (freedom 1). Access to the source code is a precondition for this ‘freedom to help yourself’ (Stallmann, 2002: 166).

- The freedom to redistribute copies either gratis or at some cost (freedom 2).

- The freedom to improve the program and release modified versions, so that the community can benefit from changes (freedom 3). Access to the source code is a precondition for this.

Nevertheless some restrictive rules are acceptable, as they can protect these essential freedoms instead of representing a threat to them. Around 1985, Don Hopkins, a well-known programmer, inspired Stallman by sending him a letter including the sentence ‘Copyleft—all rights reversed’ (Stallmann, 2002: 23). Four years later, Stallman created a copyleft license called GPL (General Public License). It was the first example of copyleft and today it is used for most FOSS projects, even if there are also other kinds of licences used in more specific conditions, as for example the Creative Commons. Copyleft is completely different from proprietary software licenses, because it ‘uses copyright law, but flips it over to serve the opposite of its usual purpose: instead of a means of privatizing software, it becomes a means of keeping software free’ (Stallmann, 2002: 22). As a matter of fact, copyleft prevents users from adding restrictions to a Free and Open Source Software program, whose modified versions must be free as well. The central idea is that ‘anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it’ (Stallmann, 2002: 91). Therefore, the freedom of future versions is preserved.

2.3. **Free Software vs. Proprietary Software**

The sale of the first IBM computer as well as the development of Unix operating system at AT&T/Bell Labs in the 1960’s marked the origins of proprietary software (The Open Group, 1995-2012). During the 1980’s, Microsoft Windows and Apple Macintosh established themselves as the main operating systems on the market (Tanenbaum and Bos, 2014). Microsoft also implemented particular software programs and database management systems for companies and institutions (Parker, 2011). The operating systems and software applications mentioned above are some examples of proprietary software, since their
source code is not publicly available.

Proprietary software is also known as closed source software and, in general, it refers to programs released under strict licences, which prohibit copying or redistributing and cannot be sold. Furthermore, even if users buy these software applications, they cannot modify them, because software companies only provide customers with the compiled and executable version of the program, in other words a black box that includes files of computer commands expressed in numeric values, instead of the source codes written by programmers (Stallmann, 2002: 126). In this way, software programs can only be modified by their companies, which benefit from any release by adopting the view of ‘let’s keep it a secret so we can make money off of it’ (Greenlee, 2007). Proprietary software programs can be commercial or free, that is to say that users do not necessarily have to pay for them. For example, Microsoft’s web browser, Internet Explorer, can be downloaded for free, but it is closed source, as users only have the right to use it at no cost. Therefore, just as observed for free software, the price does not always determine the kind of software application.

The FOSS movement believes in ‘a world where freedom and knowledge survives the compiler’ and it is against proprietary software, because, to use Stallman’s words, ‘the system that says you are not allowed to share or change software is antisocial . . . unethical’ (Stallmann, 2002: 13, 18). Proprietary software companies defend themselves from this kind of charges with the following arguments:

- Useful software programs can be created thanks to a company power. However, the FOSS movement demonstrated that ‘a network of passionate, geeky volunteers could write code just as well as the highly paid developers at Microsoft’ (Howe, 2006). In fact, even though a lot of users contributing in FOSS projects are not professionals, they are able to take good decisions and to solve any kind of problem, because the wisdom of the crowds is usually better than experts’ individual knowledge (Surowiecki, 2004).

- Copyright protects intellectual property. According to Stallman (Stallmann, 2002: 122), programmers should act like ‘the great artists and artisans of medieval times, who didn’t even sign their names to their work’, because the work itself and its aim are more important than authorship. In this sense, FOSS communities consider intellectual property as a limit to creativity and software development: a closed source program, for example, cannot be comprehensible and accessible to everyone,
because users are not allowed to localise it.

- Working without a reward is impossible, and incomes are also indispensable for software development. The FOSS philosophy does not condemn personal gain, but proprietary software as a destructive means to obtain it, ‘because the restrictions reduce the amount and the ways that the program can be used’ (Stallmann, 2002: 38). As observed by Stallman, there are some challenges which cannot be refused by the FOSS movement, like for example secret hardware, non-free libraries, software patents and lack of free documentation. According to the computer scientist (Stallmann, 2002: 30) this latter, in particular, is ‘[t]he biggest deficiency in free operating systems’, since free manuals are essential for FOSS packages. As for Free and Open Source Software, all users must have the freedom to modify the manual and make copies so that it can be adapted to new versions of the program and redistribute.

Furthermore, proprietary software companies usually debase Free and Open Source Software by arguing that this kind of programs lacks reliability and security. But in his letter to Microsoft, Peruvian Congressman Villanueva (2002) explains why FOSS programs are even more secure than proprietary software:

‘To guarantee national security or the security of the State, it is indispensable to be able to rely on systems without elements which allow control from a distance or the undesired transmission of information to third parties. Systems with source code freely accessible to the public are required to allow their inspection by the State itself, by the citizens, and by a large number of independent experts throughout the world. Our proposal brings further security, since the knowledge of the source code will eliminate the growing number of programs with *spy code*.’

Villanueva underlines how the autonomy from companies, which can only be guaranteed by FOSS programs, is important within organisations and governments. In this way, FOSS applications also increase the efficiency of the latter, because, since source code is available, everyone can rapidly fix bugs or add new features. According to Villanueva, it is indeed impossible to demonstrate the superiority of proprietary software security, because it should be done with ‘the public and open inspection of the scientific community and users in general’, and the actual guarantee ‘is based only on promises of good intentions (biased, by any reckoning) made by the producer itself’. However, someone could argue that exactly for the same reason we cannot determine whether proprietary software is substantially worse than FOSS programs.
Indeed quality cannot be guaranteed in both cases, but we do know that FOSS users can at least evaluate it more directly. Despite its achievements, as stated by Driver (2005), Free and Open Source Software ‘won't destroy industry giants, such as IBM and Microsoft, but it will revolutionize software markets by moving revenue streams from license fees to services and support’.

2.4. Free Software vs. Open Source

As the meaning of Free Software could be misunderstood, in 1998 a group of people in the community decided to replace it by Open Source Software (Stallmann, 2002). The two terms denote nearly the same kind of programs, but they became associated with two separate movements sharing different philosophies and values: the Open Source Software movement and the Free Software movement.

The Open Source Initiative (OSI) defines Open Source software by the following criteria:

1. Open Source software can be freely redistributed.
2. Open Source software can be distributed in source code, which must be available, or compiled form.
3. Users are allowed to modify Open Source software and to distribute its modified versions under the same terms as the original license.
4. The license can prevent users from distributing modified versions of the source code only if patch files are provided in order to make changes during build time.
5. Everyone, without discriminations, is allowed to use, distribute and modify Open Source software.
6. Open Source software can be used in any field of endeavour.
7. Users who receive a redistributed version of Open Source software do not need an additional license, as the rights attached to the program are automatically applied to them.
8. The license must not be specific to a product, but to the original software distribution.
9. The license must allow other programs than Open Source software to be distributed on the same medium.
10. The license must be technology-neutral.
By definition, Open Source and Free programs are very similar: users have access to the source code and they are allowed to modify and redistribute it. But although almost every Open Source software program is also Free Software, the first only promotes the availability of source code for the collaborations and innovations it can produce. Freedom only involves practical benefits and it is not an ethical question for the Open Source movement, as a matter of fact this latter rejects users’ right to ‘share with their neighbor’ and to be free from the companies’ control (Stallmann, 2002: 169). For this reason, some licenses, considered as restrictive by the Free Software philosophy, are accepted by the Open Source movement instead. As underlined by Stallman (Stallmann, 2002: 57), ‘[o]pen source is a development methodology’, whereas free software is the solution to a social problem, as the Free Software movement believes that ‘freedom, not just technology, is important’. Since the difference between the two movements does not concern practical reasons but basic values, they often work together on some projects, because, to use Stallman’s words, the Free Software community does not ‘think of the Open Source movement as an enemy. The enemy is proprietary software.’

2.5. Conclusion

This chapter was focused on Free and Open Source Software. After describing the origins of the Free Software community, we have discussed the controversial definition of the term free and we have specified the characteristics and aims of Free and Open Source Software applications (section 2.2.). In section 2.3., we have outlined the origins of proprietary software programs, we have also described them in opposition to FOSS applications, while highlighting the negative aspects of closed source codes. In section 2.4., we have firstly provided a definition of Open Source Software. Then, we have explained that the main difference between Open Source Software and Free Software programs concerns the philosophy and values of the two separate movements they are associated with.
3. FOSS Localisation

‘Every decision a person makes stems from the person’s values and goals. People can have many different goals and values; fame, profit, love, survival, fun, and freedom, are just some of the goals that a good person might have. When the goal is to help others as well as oneself, we call that idealism.’

(Stallman Richard M., Copyleft: Pragmatic Idealism)

3.1. Introduction

As mentioned before, with the development of Web 2.0, users became active contributors of content production. Although ‘[a]t one level, all Internet content is user-generated, as the medium by its very nature promotes interactive, many-to-many modes of communication’ (Flew, 2008: 35), this community-based approach is especially employed in FOSS projects where users are engaged in remote, collaborative work.

This chapter focuses on FOSS Localisation including its workflow and technical features (section 3.2.), such as tools and file formats (section 3.3.). It then analyses the main aspects of this kind of collaborative work and discusses most users’ motivations (section 3.4.). The chapter finally describes some examples of products that are being localised by the FOSS community (section 3.5.).

3.2. Localisation Workflow

In general, FOSS communities are not very hierarchical, but, according to Ruffatti (2008), they are usually characterised by a clearly-defined organisational structure including the following roles:

- Developers, who usually create FOSS applications and discuss suggestions for improvements.
- Contributors, who send codes, patches and documentation. They are allowed to make changes and fix eventual bugs.
- Project Owner, who assigns administrator permissions to contributors. He is also responsible for the release of new versions and the stabilisation of patches, new
codes and changes implemented by contributors in development builds.

- Project Board, who supports the project owner and guarantees conflicts resolution. He also ensures that projects remain faithful to their purpose.

Passionate volunteers typically contribute in different fields of FOSS projects, such as programming, design and localisation, in order to create or improve programs and follow their progress (Wolff, 2011).

Although, in general, localisation projects have different release cycles depending on their contents, languages and target audiences, localisation workflows are characterised by some common stages (Schäler, 2009). From this point of view, FOSS localisation is not so different from conventional software localisation and, according to Wolff (2011), its workflows mostly consist of the following processes:

- Analysis of the localisable software and available resources in relation to its size and characteristics.
- Review of the source text.
- Extraction of translatable text from the source code and creation of a translation template, from which new translations are generated.
- Translation of the file and creation of terminology database.
- Automated testing, manual testing or in-product testing. In FOSS communities, workload is shared among localisers, so they generally review one another’s translations (Wasala et al., 2013). Automated testing consists of software review by means of quality checks, which find linguistic or technical errors and indicate possible solutions to fix them. In manual testing, translations marked as needing work are analysed by reviewers, who simply remove the marker to approve them. Finally, in-product review consists of testing software through the translated interface: all those elements involved in the users’ experience are examined, such as error messages, boxes size, accelerators, links, documentation – which is usually written by means of Wikis – and online help. In all the above mentioned cases, when testing is completed, bug reports are created and tracked in a web-based system in order to ensure software quality.
- Submission of translations by individual users working alone, who simply have to send them to a developer, or by groups translating several languages at the same
time. Most FOSS programs are developed with a version control system to optimise and facilitate files uploads and downloads. Thanks to this system, all submitted translations can then be used as a basis for future versions of the software.

- Promotion of the software in various countries, sometimes by participating to events.

Since the study conducted for this thesis especially focuses on beginners in FOSS localisation projects, we have finally listed below some key rules that Wolff (2011) addresses to these users:

1. **Start small.** A localiser must choose tasks depending on his potential available time and resources.
2. **Read the instructions.** They can be helpful to better localise the software.
3. **Stay consistent.** Just like for all other kinds of translation, software localisers must ensure terminology and style consistency.
4. **Be careful, but be bold.** Localisers must be careful while translating and they should make the appropriate researches, but at the same time they must have a lot of initiative.
5. **Expect criticism.** Constructive criticism is very useful, since it leads to software improvements and resolution of errors.
6. **Enjoy it.** FOSS projects are usually carried out by volunteers with a strong community orientation who enjoy localising and discussing about software issues. A more direct communication takes place through chat channels, which provide contributors with immediate feedbacks to any question or problem, mailing lists, one or several systems par project allowing group emails exchange, and forums, which are rarely used for discussion.
3.3. Tools

Kingscott (1988: 14) briefly retraces the history of translation tools affirming that:

‘In the distant past translators, like authors, produced their work writing it out by hand, starting with a draft and rewriting it. Then came the portable typewriter which was widely taken up by translators. The next technological advance was the dictating machine, which was adopted by in-house translation services and by some of the higher output translators who in turn used hired typists to actually produce the written text. And then came word processing.’

Personal computers, Computer-Assisted Translation (CAT) tools and the Internet then followed forming contemporary workstations which are characterised by standards, translation memories, multilingual documents management systems, terminology databases, speech recognition, access to document servers and machine translation (Armstrong, 2011). All these machine supports are generally integrated into a project management tool (Quah, 2006: 13), which assists translators during the whole translation process, thus allowing to ‘control the production of high quality translation’ (Allen, 2003: 29).

Hutchins and Somers (1992) identify different degrees of human intervention and mechanisation. Fully Automatic High Quality Translation (FAHQT) corresponds to the highest degree of mechanisation, while at the opposite side we find Human Translation (HT). Then two kinds of Computer-Assisted Translation (CAT) are at the middle of the scale: Human-Assisted Machine Translation (HAMT), which gives prominence to the role of tools, and Machine-Assisted Human Translation (MAHT), which emphasizes translator’s function. Even though, these categories are not absolute, since tools are now multifunctional and constantly evolving, Hutchins and Somers’ categorization ‘remains useful as a point of reference for classifying translation in relation to technology’ (Quah, 2006: 6). A more precise classification is otherwise possible by listing tools in relation to their function (Vandenberg, 2009) and to the main processes of a typical translation workflow:

- Pre-translation involves terminology management – i.e., dictionaries, glossaries, terminological databases and automatic term look-ups are provided in order to facilitate the reuse of appropriate terminology –, machine translation and translation memories. The latter consist of ‘a type of linguistic database that is used to store source texts and their translations. The texts are broken down into short segments that often correspond to sentences’
(Bowker, 2002: 92). Source and target sentences are aligned, so that matches can be identified by the program: exact matches are automatically inserted in translations, while fuzzy matches (i.e., approximate matches) are only showed to translators. In this way, a ‘collage translation’ (Mossop, 2006: 786) is possible by recycling previous translations stored in translation memories, thus eliminating duplicate work.

- Translation Editing implies the use of word processors or speech recognition.
- Review can include quality checks and change tracking.
- Exchange involves standards to facilitate files importation and exportation.

For the aim of this study we are now focusing on FOSS localisation tools used by FOSS communities to localise programs. These applications enable contributors to be more efficient, improve projects and team management, and increase software quality. But their main advantage is that they are mostly adapted to communities’ needs: for example, some of these tools provide localisers having low English level with an alternative source language to better understand source text while localising (Wolff, 2011). According to the list in the Open Translation Tools manual (FLOSS MANUALS, 2011), there are different types of FOSS localisation tools, which can be grouped in the following categories:

- Offline or online editors, such as Pootle, Virtaal, Poedit and Lokalize, which are compatible with PO, XLIFF and related formats.
- Project management tools, including for example Transifex, Pootle, Launchpad Translations and World Wide Lexicon, which control tasks, users’ roles and permissions, and provide all information about projects. They are usually compatible with other localisation tools and version control systems.
- Machine translation tools, like Moses and Apertium.
- Computer-assisted translation (CAT) tools, such as OmegaT, Virtaal and QT Linguist, which can be used to support localisers working directly on the source code or translating the source text previously extracted from it (Translate, 2013a).
- Dictionaries and glossaries, including CollaboDict and GNOME Dictionary, where localisers can find specific terms and meanings in various languages.
- Wiki translation tools, like Cross Lingual Wiki Engine and Betawiki, which consist of modules and extensions for the translation of wiki contents.
- Tools for subtitling, for example GNOME Subtitles, which allow users to insert subtitles in videos.
- Code Libraries and Packages, including the Open Translation Engine and the Translate Toolkit.

Integration and interoperability among tools are essential for the localisation process, where, as pointed out by Morado Vázquez and Wolff (2011), localisers have to deal with several formats in constant development. Although FOSS communities often develop toolkits to avoid compatibility problems, they rarely manage to use exclusively these sets for the entire localisation process (FLOSS MANUALS, 2011). In a context of multiple formats, standards become indispensable for completing projects more rapidly and ensuring software quality. Software companies mainly employ XLIFF, the XML Localization Interchange File Format, which contains several bilingual files including a header, specifying the metadata of the document, and a body, which consists of translation strings divided into source text and target language (Morado Vázquez and Wolff, 2011). Even though the XLIFF format is often used in FOSS CAT tools (Cánovas and Samson, 2008), for most FOSS projects, localisers work with Gettext. This system consists of code internationalisation and a subsequent extraction of translatable text from the source code generating a text file known as POT file (Portable Object Template). This latter is sent to the administrators of the different projects who will create a PO (Portable Object) file (i.e., a bilingual text file) for each target language, then grouping all the files in a directory called po, locales or lang (Wolff, 2011). The PO format is very similar to XLIFF, since it contains both source and target strings, and despite the presence of several other formats, such as Qt Files, Java Properties and PHP, it is the prevailing and most popular interchange format in the FOSS localisation world (Frimannsson and Hogan, 2005: 10).

3.4. Aspects and Motivations
The ideals of freedom and altruism are at the basis of FOSS localisation. In his analogy of recipes, Stallman (Stallmann, 2002) supposes that someone modifies a recipe and then gives this new version to his friends, who have enjoyed the meal. In this metaphor, the recipe corresponds to source code, while the people involved represent the FOSS community. The computer scientist then imagines a world where ‘recipes were packaged inside black boxes’
and you were not allowed to modify them or to provide friends with copies, because otherwise ‘they would call you a pirate’. This world, ‘in which common decency towards other people is prohibited or prevented’, represents proprietary software society (Stallmann, 2002: 159). We are now analysing some positive and negative aspects of FOSS localisation as well as some users’ motivations for this collaborative work.

Although working as volunteers in FOSS localisation might cause a loss of motivation, it is important to consider that FOSS communities provide users with the opportunity to extend their knowledge acquiring expertise in a context of authentic meritocracy, where respect and acknowledgment for the quality of ideas are more motivational than monetary rewards (Howe, 2008). Professional support is not guaranteed, but users can find all the help they need within FOSS communities, since ‘[i]here is a big culture of doing favours in FOSS projects’ (Wolf, 2011: 47) and, as observed by Surowiecki (2004: 19), ‘[t]he web provides a perfect technology capable of aggregating millions of disparate, independent ideas in the way markets and intelligent voting systems do, without the dangers of ‘too much communication’ and compromise’. Unlike hierarchical structures, this type of collaborative contexts encourages and promotes people’s freedom of expression and cooperation, as a matter of fact ‘[i]f great minds think alike – and in many circumstances they do – then they really constitute only one mind’, while ‘[a] diverse group of solvers results in many different approaches to a problem’ (Howe, 2008). Therefore the best way to obtain significant results is to allow users to localise programs or adapt them to their needs, thus guaranteeing the control of appearance and functionality. This will also preserve continuity, because, if for example the contributors of a FOSS project abandon it for some reasons, features or licences cannot be reversed and other users can always start working again on the project in the future. While, on the contrary, if a company decides to stop producing a particular proprietary software program, it might not be licensed anymore.

FOSS localisation projects are carried out by users who want to make software programs and documentation accessible to everybody (Wolf, 2011). FOSS localisers become “remediator[s]” of linguistically inaccessible products and “direct producer[s]” of translation on the basis of their knowledge of the given language (O’Hagan, 2009: 97). In this sense, localisation concerns access to information and it is above all a social matter, since about half of 6,000
spoken languages, as well as the cultures, collective memory and values they represent, risk disappearing (UNESCO, 2011). Some FOSS communities, like for example Translate.org.za\(^{12}\), aim to preserve minority languages and promote users’ mother tongues by translating new information into a particular language, since products localisation also spreads different cultures’ ideas and beliefs (FLOSS MANUALS, 2011). Another significant aspect of these communities’ work is its effect ‘on the lives of the people they translate for, on their economic well-being, on their access to justice, on their environment and on their health’ (O’Brien and Schäler, 2010). Access to information in a comprehensible language is, in fact, ‘a basic "linguistic" right based on a fundamental human right’ (De Varennes, 2001: 19) and in some situations it can be a ‘life-saving and community-sustaining service’, like for example South African HIV/AIDS information content for communities (FLOSS MANUALS, 2011: 11). In 2002, during his speech at a meeting of the United Nations Information and Communication Technologies Task Force, Kofi Annan stated that:

\begin{quote}
‘The new information and communications technologies are among the driving forces of globalisation. They are bringing people together, and bringing decision makers unprecedented new tools for development. At the same time, however, the gap between information "haves" and "have-nots" is widening, and there is a real danger that the world’s poor will be excluded from the emerging knowledge-based global economy.’
\end{quote}

Therefore peoples’ access to knowledge not only is a way of preserving languages, but also the first step to development for countries’ economies. As a matter of fact, it could stimulate local content creation and software companies’ development, thus improving governments’ administrations which will not depend on proprietary software anymore. Local companies will start localising software according to national needs (FLOSS MANUALS, 2011) and they will then export local products to foreign markets. This will encourage companies across the world to develop commercial FOSS programs instead of proprietary software (Stallmann, 2002).

Quoting Vaclav Havel, Stallman (Stallmann, 2002: 129) advises users to ‘work for

\(^{12}\) Translate.co.za is a non-profit organisation, whose mission is ‘producing Free and Open Source software that enables and empowers South Africans’ (McKay, 2005). It was founded in 2001 by Dwayne Bailey to localise FOSS programs in the eleven official South African languages (Wolff, 2006).
something because it is good, not just because it stands a chance to succeed’, thus rejecting proprietary software which ‘is not what is good for society’. As mentioned above, FOSS localisation is above all a social issue, therefore FOSS users’ motivations can be associated to those ones of traditional social movements, where Klandermans (1997) distinguish three kinds of motives: collective motives, depending on the contributors’ opinion on the movement and its aims, social motives, mainly based on the other people’s reactions, and reward motives, resulting from the estimation of benefits and costs. In their study on FOSS communities’ motivations – which was based on a web survey conducted among developers of FOSS projects hosted on SourceForge.net –, Lakhani et al. (2002) identify four different groups of users:

- Stimulated learners, representing 29% of the participants, who are motivated by their skills improvement and fun.
- Hobbyists (27%), who feel as part of the hacker community and want to develop their skills for non-work activities.
- Professionals (25%), that is to say experienced programmers who are interested in improving their work skills and establishing their status in FOSS communities.
- Community Believers (19%), who are strongly inspired by the FOSS communities’ values and goals.

According to Lakhani et al. (2005: 17), the results of this study indicate that the FOSS community is a big tent, where contributors’ motivations are related to a ‘combination of intrinsic and extrinsic factors13 with a personal sense of creativity being an important source of effort’. Users contribute to FOSS localisation projects for different reasons: some of them would like to be engaged in technical translations (mission-oriented) or support humanitarian causes (subject-oriented) (Bey et al., 2006), while others are just interested in the translation

13 Quoting Ryan and Deci (2000: 56), Lakhani et al. (2005: 17) define an intrinsic motivation as ‘the doing of an activity for its inherent satisfactions rather than for some separable consequence’. That is to say that people are motivated by challenges or their fun rather than ‘external prods, pressures, or rewards’. At the opposite side, extrinsic motivation is based on ‘immediate and delayed payoffs’. In the context of FOSS communities, these payoffs might include monetary rewards or users’ interest in a particular software program (Von Hippel, 2001).
of a particular attractive software program (O’Hagan, 2009), some contributors simply enjoy localising, while some others would like to fix software bugs (Wolff, 2011). But despite this variety of motivations, according to Stallman (2002: 54), FOSS users should have a unique common aim:

‘What we need above all, to confront any kind of challenge, is to remember the goal of freedom to cooperate. We can’t expect a mere desire for powerful, reliable software to motivate people to make great efforts. We need the kind of determination that people have when they fight for their freedom and their community, determination to keep on for years and not give up.’

3.5. Success Stories

As discussed above (section 3.4.), unlike large multinational companies, who only localise for people representing a business, the FOSS community provides everyone with access to information technology thanks to its thousands of contributors in FOSS localisation (O’Brien and Schäler, 2010). In this section, we are focusing on FOSS success stories, describing in particular two projects: Apache OpenOffice and Mozilla.

OpenOffice is a FOSS desktop application developed from the source code of StarOffice, a proprietary software program belonging to Sun Microsystems. In 2002, OpenOffice.org 1.0 was released. But when Sun Microsystems was acquired by Oracle Corporation in 2010, numerous volunteers abandoned the project and created The Document Foundation in order to ‘fulfil the promise of independence written in the original charter’ (The Document Foundation, 2010-2013: 376). A new program based on OpenOffice was thus released in January 2011: LibreOffice. Afterwards, the Apache Software Foundation (ASF) obtained OpenOffice.org which became a ‘top-level project’ in 2012 (The Apache Software Foundation, 2011-2012). Apache OpenOffice is a vast FOSS project involving over 750 contributors and an international community of almost 400,000 users (The Apache Software Foundation, 2011). The suite is used by tens of millions of people and it is considered as the FOSS equivalent of Microsoft Office, as it is composed of the following six applications: a full-featured word processor (including its web-authoring tool), multimedia presentations, illustration, spreadsheet, database and equation editor (The Apache Software Foundation, 2011-2012). Apache OpenOffice is supported by different operating systems, such as Windows, Solaris, Linux and Macintosh, and it produces documents in a standard format, thus avoiding compatibility problems and ensuring
interoperability with other common packages, like for example Microsoft Office. The Apache Software Foundation localisation project, which is hosted on translate.apache.org, its official Pootle Translate server, concerns over 110 languages.

Another example of FOSS success story is represented by the Mozilla project, which had its origins in 1998, when the Netscape Communicator\textsuperscript{14} source code was released (Mozilla.org contributors, 1998–2015). The project promoted openness and innovation in the browser market and provided contributors from around the world with the opportunity to develop their creativity on the Internet. The original mission was the development of the Mozilla Application Suite, but in 2003 contributors abandoned this objective to focus on the creation of different tools (Mozilla.org contributors, 1998-2009). In 2004, the community released its most successful product: Mozilla Firefox, a web browser which reached over 100 million downloads in less than a year (Mozilla.org contributors, 1998–2015). Since then, Mozilla Firefox has become very popular and it is now supported by various operating systems, such as Windows, Linux, Mac OS X. Firefox localisation project concerns over 20 thousand words for the tool itself, and then documentation and marketing materials (Wolff, 2011). The Mozilla localisation project is hosted on localize.mozilla.org and it involves nearly 400 L10n teams who translate using Pootle interface.

3.6. Conclusion

This chapter was based on FOSS Localisation. We have firstly described some common stages of FOSS localisation workflows (section 3.2.). In section 3.3., we have outlined the origins of translation tools and we have then focused on tools and file formats used in FOSS localisation projects. In section 3.4., we have analysed the main aspects of FOSS localisation and most users’ motivations. We have finally provided some success stories, describing some products developed and localised by the FOSS community (section 3.5.).

\textsuperscript{14} Netscape Communicator is an Internet suite released in 1997. It is characterized by the following applications: Netscape Navigator, Netscape Messenger, Netscape Collabra, Netscape Address Book, Netscape Composer, Netscape Netcaster, Netscape Conference and Netscape Calendar (Janalta Interactive Inc., 2010-2015).
4. Evaluating and Comparing

‘Quality is not a single, unchanging characteristic of software: it means different things to different people at different times of a product’s life cycle.’

(King M. and Maegaard B., Issues in Natural Language Systems Evaluation)

4.1. Introduction

In this chapter we will delineate the approach to be taken for Virtaal and Pootle’s evaluation as well as for the comparative study of these two tools. First of all, we will provide an overview of past evaluation methods (section 4.2). Then, a methodology for our study will be defined following the EAGLES 7-step recipe (section 4.3), and criteria will be specified on the basis of 6 key quality characteristics defined by the EAGLES final report (section 4.4). In section 4.5, we will describe the kind of tests used for the evaluation.

4.2. Past Evaluation Methods

Researchers, developers and translators usually conduct studies for different purposes, often publishing their reviews on specialised magazines (Quah, 2006). There are no universal methods that can be adapted to specific evaluations, because, the variety of available translation tools, their frequent updates and the different goals defined for each study make the construction of uniform methodologies more difficult (ibid).

By the 1990s, some researchers started trying to build a universal evaluation method. In particular, Margaret King succeeded in determining a procedure which would then be at the basis of the development of an evaluation methodology for CAT tools. It consisted of four steps (King, 1993):

1. First of all, researchers must determine the type of evaluation, and the structure of this latter must be established; then the evaluated system and users’ profile must be described.
2. In relation to each evaluation, researchers must choose the most appropriate characteristics from a predefined list. These ones should then be tested in the
concerned system.

3. For each evaluation, researchers must also choose some valid and reliable sub-characteristics and measures depending on the specific circumstances.

4. Researchers must finally determine how to obtain the results in relation to the type of evaluation, its criteria and measures.

As mentioned above, this four-step procedure is not a universal methodology of evaluation, but it is a starting point: researchers can, in fact, freely choose the characteristics, criteria or measures they prefer in relation to the evaluation they are conducting and its purpose. King’s evaluation methodology was followed and developed by the ISO 14598 standard – published to complete the ISO 9126. Afterward, various projects were carried out in this field, such as the International Standards for Language Engineering (ISLE) and the Expert Advisory Group on Language Engineering Standards (EAGLES) (Quah, 2006). This latter was launched by the European Commission in 1993 and included a research group on the development of a standard for evaluation. In particular, the initiative aimed ‘at developing a general methodology for the design of evaluations which would be applicable to all evaluations in the field of language engineering, whether they be of systems, of products or even of projects’ (King and Maegaard, 1998: 225). Different EAGLES reports were published, and in the final one an evaluation framework was designed (EAGLES, 1996). It consists of three main elements – characteristics, sub-characteristics and testing methods – and combines them with users’ needs (Quah, 2006). However, despite its universality, the EAGLES evaluation framework is not an absolute methodology, since it has been designed as a guide (Hovy et al., 2002), which, together with the ISO 9126 standard, provides researchers with useful instruments for software evaluation, including a list of criteria, different types of tests and a simple procedure to design and conduct them (Quah, 2006).

As seen above, this methodology is focused on end-users and its framework can be adapted by researchers, who can freely choose the characteristics to be evaluated and the criteria to be applied. Since our study is supposed to analyse users’ reaction to Pootle and

---

15 The ISO 9126 standard was published in 1991. It describes six key quality characteristics to be considered in the evaluation of software products, thus aiming at ensuring the quality of the latter (Quah, 2006).
Virtaal, we are going to apply the EAGLES evaluation methodology, adapting criteria to the characteristics that are going to be tested. In order to make a comparison of Pootle and Virtaal, the same criteria will be used, the same characteristics and eventually sub-characteristics will be evaluated and the same tests will be conducted for each tool.

4.3. Methodology

In 1999, the EAGLES working group published ‘*The EAGLES 7-step recipe*’, a document summarising the EAGLES methodology described in the final report into 7 key steps ‘*necessary to carry out a successful evaluation of language technology systems or components*’ centred on end-users (EAGLES, 1999: 1). Since our study is designed on the basis of this procedure, we are now going to describe it in more details.

The EAGLES 7-step recipe is characterised by the following stages (EAGLES, 1999):

1. Definition of evaluation design and requirements, i.e. its purpose, object (a system, a component or more) and context.
2. Elaboration of a task model, which identifies all relevant roles and agents, in order to delineate the users’ profile and the context of use of the system.
3. Definition of the top level quality characteristics, which implies a selection of the features to be evaluated.
4. Evaluation specifications. At this step, if there is not a valid and reliable way of evaluating the system performance in relation to the selected features, they should be broken down into sub-characteristics until they become measurable.
5. Definition of the metrics and methods to be applied for the requirements produced in step 4. For each comparable characteristic, researchers should determine the scores for a satisfactory or unsatisfactory performance, the degree of importance, and the limits in relation to the task model.
6. Evaluation design. At this stage, researchers have to prepare test materials and define when, where and how the evaluation will be performed and by whom.
7. Evaluation execution and conclusion. Researchers execute the evaluation and collect data by applying the predefined metrics and scores. The results are then presented in an evaluation report.
4.4. Criteria

By following the guidelines of the EAGLES final report, our study aims at evaluating and comparing two tools, considering the needs of some particular users, i.e. non-experienced translators. In relation to these needs, we are going to define some criteria for our study which must be as measurable as possible in order to evaluate the systems in a valid and reliable way, just as specified at the stages 3 and 4 of the EAGLES 7-step recipe.

The ISO 9126 standard was the starting point for the EAGLES working group (King and Maegaard, 1998) and it will provide the requirements to be tested in our evaluation. By the following list, we are now going to describe the six key quality characteristics at the basis of the ISO 9126 standard (EAGLES, 1996):

1. **Functionality** consists of a set of attributes measuring the presence of some functions which satisfy stated or implied users’ needs under specific conditions.

2. **Reliability** includes a set of attributes to test the capability of a system to constantly maintain its level of performance under specific conditions.

3. **Usability** involves a set of attributes to test the quality of users’ experience during their interaction with the system in question. In particular, it evaluates intuitiveness, that is to say the effort needed to learn how to use a program, users’ error frequency and severity, as well as their satisfaction.

4. **Efficiency** implies a set of attributes evaluating the time behaviour of a determined system, as well as its performance considering the existing resources.

5. **Maintainability** indicates a set of attributes testing users’ effort to adapt a system to their needs by making some corrections and improvements.

6. **Portability** denotes a set of attributes evaluating the capability of a system to be integrated in different environments.

As described in the fourth step of the EAGLES evaluation procedure, the features criteria have to be broken down into sub-attributes until they become measurable and comparable. A clear scheme of the six key quality characteristics listed above is shown in Figure 5, where each criterion is divided into a set of sub-characteristics.
Obviously, this list is neither absolute nor exhaustive. As a matter of fact, other characteristics or sub-characteristics can be taken into consideration, like, for example, **customizability** – that is to say *‘the ability to modify a product in order to satisfy a particular customer's needs’*, which was subsequently added to the list by the EAGLES working group (Rico, 2001: 5). According to Quah (2006: 134), *the evaluation criteria that interest end-users include the ‘hows’ and the ‘whats’*, which are characterised by the following questions, respectively:

- How easy is it to operate a tool? How user-friendly is a tool? How long does it take to learn? How compatible is it with other hardware and software applications? How good is the design of the working environment (the layout of the interfaces and display of windows)? How good is the support for Latin and non-Latin based languages? How easily can a tool be extended or upgraded?
- What is the processing speed? What are the linguistic capabilities? What is the

---

16 This sentence from *Translation and Technology* is a reference to Trujillo (1999: 254).
required operating system? What is the performance reliability? What are the costs and benefits?

In the table below, we have applied the above guidelines to our study: we have selected the characteristics to be analysed and evaluated specifying the sub-characteristics, and then we have associated them with some *hows* and *whats*. Since not all the questions listed by Quah (2006: 134) were suitable for the characteristics and sub-characteristics to be tested in our study, we took into consideration only some of them and we added some new ones when necessary.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Sub-characteristics</th>
<th>Example of questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>Suitability</td>
<td>What are the functions of the tool? How appropriate are they?</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>How accurate is the tool?</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Time behaviour</td>
<td>What is the processing speed?</td>
</tr>
<tr>
<td>Usability</td>
<td>Understandability</td>
<td>How user-friendly is the tool?</td>
</tr>
<tr>
<td></td>
<td>Learnability</td>
<td>How long does it take to learn?</td>
</tr>
<tr>
<td></td>
<td>Operability</td>
<td>How easy is it to operate the tool?</td>
</tr>
<tr>
<td></td>
<td>Attractiveness</td>
<td>How satisfied are users with the tool? What are the advantages/disadvantages for users?</td>
</tr>
</tbody>
</table>

Table 3: Examples of questions related to the evaluation criteria.

As shown in the table above, this evaluation will be focused on three main quality characteristics, which are the most appropriate for its purpose and context: functionality, efficiency and usability. For each of them, we have selected one or more sub-characteristics to be tested (EAGLES, 1996):

- Functionality is divided into **suitability** – that is to say ‘the presence and appropriateness of a set of functions for specified tasks’\(^{17}\) – and **accuracy**, which consists of ‘attributes of

---

software that bear on the provision of right or agreed results or effects\textsuperscript{18}.

- Efficiency refers to \textit{time behaviour}, in other words, the processing speed and throughput rates of a tool in performing its function\textsuperscript{19}.

- Usability involves \textit{understandability}, defined as ‘users’ effort for recognizing the logical concept and its applicability\textsuperscript{20}, \textit{learnability}, measuring users' effort to learn how to use the tool\textsuperscript{21}, \textit{operability} – that is to say ‘users’ effort for operation and operation control\textsuperscript{22} – and \textit{attractiveness}, evaluating users’ overall satisfaction.

4.5. Tests

As mentioned above (\textit{section 4.1.}), King (1993) suggested that researchers must choose some valid and reliable measures for their evaluation, depending on the specific circumstances. The EAGLES final report developed this step of the evaluation process by providing some testing methods, in which test types, instruments and materials were defined. In the context of our study, we will conduct a user-oriented evaluation, since of all the methods ‘it is user testing that is to some extent directly applicable to the evaluation of translators’ aids’ (Höge, 2002: 127).

The EAGLES final report defines three different types of test: \textbf{Feature Inspection, Systematic or Benchmark Test} and \textbf{Scenario test}. The first one analyses the technical characteristics of a program. It is appropriate for comparisons, since it shows the differences between systems, as well as the positive and negative aspects of preferring a program rather than another one (Quah, 2006). This test consists of a checklist of features, such as supported formats, interface languages, shortcuts and assistance tools, and it is characterised by a boolean metric, that is to say that researchers indicate whether a function is present or not. Information about the system and its functions can be collected by analysing the program or by reading user documentation (Höge, 2002). However, an important condition has to be fulfilled in order to guarantee the validity of the evaluation:

\textsuperscript{18} ISO 9126: 1991, A.2.1.2.
\textsuperscript{19} ISO 9126: 1991, A.2.4.1.
\textsuperscript{20} ISO 9126: 1991, A.2.3.1.
\textsuperscript{21} ISO 9126: 1991, A.2.3.2.
\textsuperscript{22} ISO 9126: 1991, A.2.3.3.
each checklist must be standardised, that is to say that ‘it should be applicable for any such tool and the results should be independent of situational variables’ (EAGLES, 1996: 140). The **Systematic or Benchmark test** aims at evaluating the performance of a determined system under specific conditions (EAGLES, 1999). It is unequivocally objective, since the evaluation is not influenced by subjective elements, like for example users’ involvement. The **Scenario Test** evaluates the quality of a system, but also users’ satisfaction in relation to their experience with it (EAGLES, 1999). Researchers create a sort of working environment, where some particular users perform a standardised task using a determined system (EAGLES, 1996). As human involvement increases the risk of unreliable tests, ‘it is important that a representative number of users participate’ and that they have very similar backgrounds (Höge, 2002: 134). The main difference between the scenario and the systematic test concerns users’ involvement into testing, which is typical of the first one (Höge, 2002).

In the context of this evaluation, we will perform two of the three tests described above. **Feature Inspection** will provide detailed information about Virtaal and Pootle, specifying the presence/absence of features in each program, in order to introduce the comparison of the two FOSS localisation tools (section 5.5.3.). The **Scenario Test** will let us evaluate Virtaal and Pootle by taking both an objective and a subjective approach: the first one implies the video recording of participants’ actions on screen, while the latter consists of collecting users' opinions about the tools.

### 4.6. Conclusion

This chapter was focused on the approach that we will take for the comparative study of Virtaal and Pootle. We have firstly analysed past evaluation methods (section 4.2.). Then, we have delineated a methodology based on the EAGLES 7-step recipe (section 4.3.). In section 4.4., we have specified the criteria for our study on the basis of 6 key quality characteristics defined by the EAGLES final report, while in section 4.5., we have described the tests used for the evaluation.
5. Evaluation Design and Requirements

5.1. Introduction
This chapter will be an introduction to the evaluation process. First of all (section 5.2.), we will explain the purpose of this study, then the users’ profile will be outlined (section 5.3.), analysing some personal data such as their education, computer skills, language combination and experience with localisation tools. In section 5.4., we will list and describe all the materials used for this study: information and pedagogic documents, instructions, questionnaires and applications (sections 5.4.1., 5.4.2.). In section 5.5., we will introduce Pootle and Virtaal, the localisation tools at the centre of this study, describing and comparing their features (sections 5.5.1.-5.5.3.).

5.2. Purpose
This study is focused on non-experienced translators’ reaction to two Free and Open Source Software Localisation Tools: Pootle, a web-based system, and Virtaal, a stand-alone tool. Despite the several differences which generally oppose an online system to a desktop application, these two FOSS tools are very similar, since they were created by the same developers. Therefore it is important to specify that the purpose of this study does not concern the mere comparison of some specific features in Pootle and Virtaal as an end in itself. But since both applications claim to be powerful tools with simple and clear interfaces facilitating the localisation process, we would like to figure out which of them would be more appropriate for a non-professional who wants to start contributing to FOSS projects.

In the context of this study, we have tried to evaluate and compare the main features of both tools from the user’s perspective by analysing the simulated localisation process of a little FOSS application from English to Italian. Participants filled in two questionnaires, one before and the other after the task. Afterward, in order to compare non-experts and habitual users’ opinions, an online survey was conducted among active users of both Pootle and Virtaal in FOSS community. But since this survey is not directly related to our study, its design, process and results will be treated in a separate chapter at the end of this thesis (chapter 8).
5.3. Participants

As mentioned before (section 4.4), two different approaches were applied during this study: for the objective evaluation, participants were asked to localise a little program and each experiment was video recorded and analysed; while the subjective evaluation was focused on questionnaires collecting participants' impressions about their task.

Two groups performed the experiments, each including 4 participants, who localised a FOSS application called NoteFly. The first group used Virtaal, while the second one used Pootle. Due to the limited number of participants, the homogeneity of background information is essential for this study. For this reason, participants’ personal data were collected in a general questionnaire. The two groups involved in this study consisted of 8 Italian native speakers, whose ages ranged from 24 to 30 years. All of them had English in their language combination – 5 of them had Proficient level and the other 3 Advanced. This was a precondition for this study, which was designed in English, in order to prevent results alteration because of misunderstandings. The participants’ education field was Translation Studies: some of them have just obtained their degrees, the others were still students, but all of them took at least a Translation Technology course and have used at least a CAT tool. Participants’ had no knowledge in the localisation field and consequently no experience in software localisation, therefore we explained them some basic concepts by means of a PowerPoint Presentation written in a very simple and understandable way, in order to provide them with the adequate skills to participate to this study. Five participants out of eight declared they did not know any FOSS program, but when we asked them to indicate the software applications they knew (Q9), all of them selected at least a FOSS application, indicating OpenOffice Writer or Firefox or even both. Therefore we infer that the participants to this study have very little knowledge of FOSS software and community.

5.4. Materials

Several materials were prepared for this study and we are going to describe them in this section. Before starting their task, participants were asked to read an information sheet (Annex I), containing a description of the study and specifications about its purpose, duration, benefits and risks. It also included the confidentiality terms and the researcher’s
contact details for any further question or concern. Participants then signed a consent form (Annex I) for the processing of personal data.

For each test of the study, a directory was created on the desktop including the following documents:

- *Introduction to Software Localisation*, a pedagogic PDF file which was presented before the beginning of the tests to provide participants with some basic principles of software localisation. The document design was based on the resources provided during classes at the Department of Translation Technology (or TIM in its French acronym), because their structure, which is normally used in teaching, is suitable for people who have no technical expertise and need an introduction to the subject.

- A general questionnaire (Annex II) aiming at collecting participants’ background information – the same for Virtaal as for Pootle. It consisted of 23 questions distributed on 5 Microsoft Word pages and grouped into four sections, i.e. personal details, education, computer skills and experience with translation tools. Participants were free to choose whether to complete the questionnaire on paper or on screen and most of them opted for the latter.

- Instructions for both Virtaal (Annex III) and Pootle (Annex IV), which were also provided in paper form. While creating these files we took inspiration from the *Lab Sheets* provided during the Localisation and Project Management classes at the Department of Translation Technology, as well as on Virtaal and Pootle’s online documentation, because their clear structure and very simple language were appropriate for participants to this study.

- For the tests with Virtaal, all PO files of the program, including the one which had to be localised (*it.po*), were provided. While for the tests with Pootle, a TXT file with the participants’ login and password to access the platform was included.

- A final questionnaire (Annexes V and VI) – very similar for Virtaal and Pootle –, which was completed at the end of each experiment in order to collect participants’ opinions in relation to their task. It was composed of 29 questions for Virtaal and 28 for Virtaal, which were distributed on 8 Microsoft Word pages and grouped into 4 sections analysing functionality, efficiency, usability and overall satisfaction. This questionnaire was completed on screen, with the purpose of allowing participants...
to write without space restrictions.

Our questionnaires were based on some general guidelines delineated in order to avoid the main biases in questions listed by Choi et al. (2005) 23:

- **Length.** Questions must not be too short, nor lengthy. In the first case, there could be misunderstandings because of missing information, while lengthy questions could annoy participants. In the context of our study, questions length ranged from 4 to 22 words and an introduction to each set of questions provided participants with clear and useful instructions.

- **Complexity.** Questions must be simple and specific. For this reason, we only asked one question at a time, avoiding, for example, double-barrelled questions, because otherwise both participants and researchers could be confused about which part of the question should be taken into consideration. In order to be as specific as possible, in some questions we preferred providing participants with the number of occurrences, using, for example, *5-6 times a week* or *2-4 times a week* rather than *often* or *rarely*.

- **Ambiguity.** If participants do not understand a question, they will answer in a wrong way, providing false information or opinions. For this reason, questions were written in a clear way, with a direct language omitting jargon and obscure words.

- **Objectivity.** Questions were formulated in an objective way, thus preventing participants from being influenced while answering.

- **Inappropriateness.** Our questionnaires only included questions relevant to the purpose of this study.

---

Concerning the type of questions, we decided to use both open-ended and close-ended. In this way, we obtained a combination of qualitative and quantitative approaches, which is, according to Bryman (1988), the ‘best of both worlds’. We began our questionnaires with the latter and then continued with more general, open-ended questions, in order to prevent the fatigue effect (Dörnyei and Taguchi, 2010: 9) and because this method seems to be appropriate when participants have no or little knowledge in a particular field (Colorado State University, 1993-2015). Closed-ended questions are characterised by a predefined set of answers, which can be dichotomous, i.e. yes/no, or multiple. We decided to use both, but we added a Further comments field or an option for Other to be completed in order to avoid a forced, imprecise choice because of the limited options, and we also opted for a vertical answer format to prevent participants from selecting a wrong option instead of their real choice (Choi et al., 2005). In the context of multiple choices, we decided to use two different scales: the Likert scale, which asked participants to indicate, for each statement, to which extent they agree or disagree (Completely agree, Agree, Slightly agree, Slightly disagree, Disagree, Completely disagree), and the format rating scale, which consisted in giving a score to some features of the tool to be evaluated. For their answers, we provided participants with a six options/points scale to avoid the so-called end aversion, which is also known as central tendency (Choi et al., 2005). For what concerns open-ended questions, participants were asked to freely express their opinions without space limits. Although open-ended questions take the time to answer them, in this context, they were indispensable, because they prevented participants from being influenced by the researcher (Foddy, 1993).

Other materials employed for the evaluation of Virtaal and Pootle included two applications, which we are now going to describe in more details.

5.4.1. NoteFly
The software localised in this study was NoteFly version 3.0.5., released in May 2013 under the GNU Library or Lesser General Public License version 3.0 (LGPLv3). It is a small but advanced application written in C# which allows users to create colourful notes and “stick” them all over their desktop.
NoteFly official website\textsuperscript{24} listed the main features of the application as follows:

- Creation of note boxes in different colours and sizes from the notification area, on the right side of the taskbar.
- A rich notes editor, which allows users to format text with different fonts options, such as bold, italic and underlined, as shown in the figure above.
- Notes classification thanks to different skin colours.
- Lists creation.
- Notes management system, which tracks all notes and allows users to remove or put them on screen.
- Notes sharing, including importation and exportation to similar programs or sending by email.
- A position control to “stick” notes to the top of any other application.
- Hotkeys to open some NoteFly functions.
- Highlighting of PHP, HTML and SQL elements.
- Plug-ins to extend NoteFly options.

In the context of this study, we decided to localise NoteFly for the following reasons: first of all, it is a FOSS application using the Gettext system, it has a very simple and intuitive GUI with few technical terms to be translated, and due to time restrictions – each experiment must not exceed 1 hour –, we needed a program with little translatable text.

\textsuperscript{24} URL: \url{http://www.notefly.org} (06.07.2014)
The application had in fact 1028 words, of which 630 had already been translated by the community.

5.4.2. BB FlashBack Express

During the evaluation process, we used BB FlashBack Express version 4.1.10, a screen recorder that allowed us to extract and analyse some data for each experiment. Thanks to its simple and intuitive interface, which is shown in the figure below, BB FlashBack Express can be also used by non-experienced users and it is freeware.

![BB Flashback Player Interface.](image)

This application is appropriate for usability testing, as it can produce unlimited high quality records by regulating file size to avoid any system performance problem (Blueberry Software, 2005-2014). Different recording options are possible, including full screen, region/window or webcam captures and sound. Once the recording process completed, the application allows users to share videos online or extract them as Flash and AVI formats, but in the context of this study we simply analysed the recorded experiments by activating cursor highlighting as well as clicks and keystrokes displaying.

5.5. Localisation Tools

In this paragraph we are going to introduce the evaluated tools describing and analysing them by means of Feature Inspection, one of the three test types suggested by the EAGLES report and mentioned in the previous chapter (section 4.4.). Check-listing of features is, in fact, appropriate to compare very similar programs, since it aims to determine different or common features and consider negative and positive aspects of each tool.

5.5.1. Virtaal

Virtaal is a desktop computer-assisted translation tool developed in Python by Translate.org.za. It is a FOSS application licensed under the GPL and ‘built on the powerful
API of the Translate Toolkit\textsuperscript{25} (Translate, 2013b). Originally conceived for FOSS localisation, it can also be used for other purposes due to its simple layout, and it is mostly employed in FOSS projects, such as LibreOffice, Mozilla Thunderbird and Firefox. The term ‘\textit{Virtaal}’ is a play on words: in Afrikaans, one of the numerous South African languages, the word ‘\textit{vertaal}’ and the expression ‘\textit{vir taal}’ are pronounced the same way, but they have different meanings – respectively ‘\textit{translate}’ and ‘\textit{for language}’ (Translate, 2013b).

Virtaal has a very simple and intuitive interface and it is designed to encourage localisers to work as much as possible with their keyboard, for example using shortcuts to access functionalities in order to increase their productivity (Wolff, 2011). Virtaal online documentation provides a list of all possible shortcuts – which are grouped into sections depending on the task they are used for. Table 3, for example, illustrates general shortcuts mainly used to access menu items.

\begin{table}[h]
\centering
\begin{tabular}{|c|l|}
\hline
\textbf{Global} & \textbf{} \\
\hline
Ctrl+O & Open a file \\
\hline
Ctrl+S & Save the current file \\
\hline
Ctrl+W & Close the current file \\
\hline
Ctrl+Q & Quit Virtaal \\
\hline
Ctrl+P & Show preferences dialog \\
\hline
Alt+Enter & Show file properties and statistics \\
\hline
F11 & Toggle full screen mode \\
\hline
\end{tabular}
\caption{List of the main shortcuts (Translate, 2013b).}
\end{table}

\textsuperscript{25}A FOSS toolkit including several ‘\textit{small utility programs for localisers}’ (Wolff, 2011).
Throughout this section, we will use the list sections included in the online documentation relating them to Virtaal’s analysed features. Virtaal’s ‘main aim is to allow anyone, regardless of experience, to translate productively without sacrificing quality’ (Morado Vázquez and Wolff, 2011: 78). Despite this, Virtaal provides experienced localisers with a powerful and feature-rich tool including:

- **Automatic Language Identification.**

- **Auto-completion**, through which Virtaal automatically complete some long words. Suggestions can be accepted by pressing **Tab**, or rejected by simply continuing the translation.

- **Auto-correction** of common typing mistakes or spelling errors. For some languages, even punctuation marks are automatically adapted.

- **Fast navigation** within strings. After typing a translation, localisers just need to press **Enter** to go to the next string, while they can move through strings with **Ctrl+Down** and **Ctrl+Up**. There are also other modes to move within translation units, which can be activated under the menu bar on the left top of the window. They include: Incomplete Mode – which consists of moving between untranslated and fuzzy strings –, Workflow Mode, depending on the strings status, Quality Check Mode and Search Mode, allowing localisers to look for words or expressions in the source or target texts. A complete list of shortcuts for Navigation can be found in the table below.

<table>
<thead>
<tr>
<th><strong>Navigation</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enter</strong></td>
</tr>
<tr>
<td><strong>Ctrl+Up</strong></td>
</tr>
<tr>
<td><strong>Ctrl+Down</strong></td>
</tr>
<tr>
<td><strong>Ctrl+PgUp</strong></td>
</tr>
</tbody>
</table>
### Navigation

<table>
<thead>
<tr>
<th>Shortcut</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+PgDown</td>
<td>Move 10 units down</td>
</tr>
<tr>
<td>Ctrl+F, F3</td>
<td>Search</td>
</tr>
<tr>
<td>Ctrl+G</td>
<td>Move to next search match</td>
</tr>
<tr>
<td>Ctrl+Shift+G</td>
<td>Move to previous search match</td>
</tr>
</tbody>
</table>

Table 4: List of Shortcuts to move within strings (Translate, 2013b).

- **Copy source text to translation** by pressing Alt+Down, which is very useful and time-saving when source strings contain numerous XML mark-ups or variables.

- **Placeables manipulation.** All special elements such as numbers, acronyms, e-mails, URLs, variables, tags, hot keys and shortcuts are called placeables in Virtaal. The tool identifies these elements and puts them in red, so that they can be easily recognised and inserted into the target string. Table 5 lists shortcuts for placeable manipulation and for other actions in the current strings.

### Units

<table>
<thead>
<tr>
<th>Shortcut</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt+Left</td>
<td>Select previous <em>placeable</em></td>
</tr>
<tr>
<td>Alt+Right</td>
<td>Select next <em>placeable</em></td>
</tr>
<tr>
<td>Alt+Down</td>
<td><em>Copy</em> the source or selected <em>placeable</em> to the target</td>
</tr>
<tr>
<td>Shift+Enter</td>
<td>Enter a new line</td>
</tr>
<tr>
<td>Ctrl+Enter</td>
<td>Mark unit with marker <em>Needs work</em> as <em>Translated</em> and go to the next unit</td>
</tr>
</tbody>
</table>

---

60
Table 5: List of Shortcuts used in strings (Translate, 2013b).

- **Web look-ups**, which allow localisers to search a word or an expression on the Internet directly from Virtaal Interface.

- **Machine translation systems**, such as Apertium, Google Translate, Microsoft Translator and Moses. Localisers can have a first automated translation by enabling various Virtaal’s plug-ins.

- **Local and external translation memories**. The first includes current file translations, while the remote database involves previous translations or web based services, like for example a TinyTM server. Virtaal’s suggestions are normally displayed under the editing area.

- **Terminology help**, which is provided to translate specific terms through local terminology files, downloaded files or Open-Tran.eu.

- **Quality check**, which alerts localisers about errors in the current string. Users can also activate the quality check mode to move within incorrect strings, which are sorted by issues. Table 6 shows shortcuts used to access the above mentioned Virtaal plug-ins.

<table>
<thead>
<tr>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+Shift+Enter</td>
<td>Mark unit as <em>Needs work</em> and go to the next unit</td>
</tr>
<tr>
<td>Ctrl+Z</td>
<td>Undo the last change</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plug-ins</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+1</td>
<td>Use the first translation suggestion</td>
</tr>
<tr>
<td></td>
<td>(use Ctrl+2, etc. for others)</td>
</tr>
<tr>
<td>F8</td>
<td>Show/Hide <em>checks</em></td>
</tr>
<tr>
<td>F9</td>
<td>Show/Hide translation suggestions</td>
</tr>
</tbody>
</table>
### Plug-ins

<table>
<thead>
<tr>
<th>Ctrl+T</th>
<th>Add a term to the local terminology file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right click on selection</td>
<td>Access external look-up features</td>
</tr>
</tbody>
</table>

Table 6: List of Shortcuts to access Plug-ins (Translate, 2013b).

In addition to all the above listed features, Virtaal can work on different operating systems and support several file formats including Gettext PO and MO, XLIFF, Qt files – such as .ts, .qph and .qm –, OmegaT glossaries, TBX, TMX and Wordfast TM (Translate, 2013b).

#### 5.5.2. Pootle

Pootle is an acronym for PO-based, Online Translation and Localisation Engine, but the name was also inspired by a character in The Flumps, a children's program broadcasted on BBC (Tectonic, 2006). It is a FOSS localisation web-based system developed by Translate.org.za and released under the GPL. It can be used as an online localisation project management tool or as a local Intranet, normally installed by administrators who also provide the tool with features, such as statistical information on the translation progress and work assignment, thus making it ideal for collaborative work. For this reason, Pootle is used in several FOSS projects, such as Apache OpenOffice and Mozilla Firefox. Sometimes users have to register to access the translation editor or use advanced functionalities. A Pootle server can host one or numerous projects and languages: some servers are used for one localisation project into many languages, while others include several localisation projects into a single language (Wolff, 2011). In a third scenario, a Pootle server can contain numerous localisation projects for different languages. In this case, the server main page will display the two categories, while the accounts’ dashboard will show localisers’ selected languages and projects. Beyond the localisation progress bar for each project and language, Pootle provide localisers with detailed statistics reporting the number of localised strings and words (Wolff, 2011).

Just as for Virtaal, on the basis of online documentation, we have listed Pootle features as follows:
- Navigation modes, including Translated, Incomplete, Untranslated, Needs work, Suggestions, My Suggestions, My Submissions, My Overwritten Submissions and Search.

- Three different translation modes: Submit, Needs work and Suggest. This latter allows localisers to provide suggestions, thus enriching localisation with different ideas for the same strings. Suggestions can then be accepted or rejected by simply clicking on respectively a green tick or a red cross.

- Terminology help in the current string, which can be general for each language, or limited to projects and languages (Wolff, 2011).

- Comments option, which allow users to comment on strings in order to communicate with other agents working on the same file.

- Alternative source languages, which are selected by localisers in their accounts settings. They are displayed above the main source language during the localisation process showing how strings have been translated into other languages. This feature is especially useful for terminology disambiguation as well as for localisers who understand other languages better than English.

- Quality check, involving different checks for errors in localisation. By activating the Checks mode, localisers just need to select the error type to move within strings with that particular problem. Errors with a link to the online documentation are eventually displayed on the top-right corner of the localisation strings in order to provide localisers with further information.

- Placeable manipulation. Localisers simply have to click on placeables to move them to the target string.

- Machine Translation systems, such as Google Translate and Apertium. They are usually displayed by an icon in the source strings. When a suggestion from the MT is accepted by a localiser, it is automatically marked as fuzzy so that it can then be reviewed.

In addition to the above listed features, Pootle allows localisers to upload or download the PO or XLIFF files to work offline with Virtaal or other desktop localisation tools. Localisers can also download ZIP files containing the entire localisation projects. For what concerns the uploaded files, they will be merged with previous translations in the database. This process depends on localisers’ permissions and it involves three different types of merging:
1. **Merge.** New translations are accepted in case of untranslated strings and become suggestions in case of conflict.

2. **Suggest.** New translations are not automatically accepted, since their submissions imply a review stage.

3. **Overwrite.** This option is only possible for localisers who have the appropriate permission and it is used to replace previous translations with new ones.

Pootle supports different file formats by means of the Translate Toolkit API. In particular we distinguish two kinds of formats: **bilingual**, which contain both source and target language – such as Gettext PO, XLIFF, Qt TS, TBX and TMX —, and **monolingual**, consisting of only one language file, such as Mac OSX strings, PHP arrays, Java properties and various subtitles formats (Translate.org.za, 2012). For the latter, the Translate Toolkit also provides localisers with format converters to Gettext PO, so that source and target strings can be stored all together in version control systems.

5.5.3. **Summary and Comparison**

We are now going to resume the results of Feature Inspection by means of Table 7 comparing Virtaal and Pootle’s features. The following check-list firstly presents some general information about the tools and is then divided into five sections: localiser assistance-tools, supported formats, shortcuts and navigation modes.

<table>
<thead>
<tr>
<th>FOSS Tool</th>
<th>Virtaal</th>
<th>Pootle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>0.7.1</td>
<td>2.5.1</td>
</tr>
<tr>
<td>Type</td>
<td>Desktop CAT tool</td>
<td>Web-based localisation tool</td>
</tr>
<tr>
<td>Implementation</td>
<td>Python/GTK+</td>
<td>Python/Django Framework</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Localiser Assistance Tools</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-completion</td>
</tr>
<tr>
<td>FOSS Tool</td>
</tr>
<tr>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Auto-correction</td>
</tr>
<tr>
<td>Machine Translation</td>
</tr>
<tr>
<td>Translation Memory</td>
</tr>
<tr>
<td>Terminology Help</td>
</tr>
<tr>
<td>Placeable Manipulation</td>
</tr>
<tr>
<td>Web Look-ups</td>
</tr>
<tr>
<td>Alternative Source Language</td>
</tr>
<tr>
<td>Translation Status Notification</td>
</tr>
<tr>
<td>Comment Option</td>
</tr>
<tr>
<td>Quality Check</td>
</tr>
<tr>
<td>Documentation/Online Help</td>
</tr>
</tbody>
</table>

**Supported formats**

<table>
<thead>
<tr>
<th></th>
<th>Virtaal</th>
<th>Pootle</th>
</tr>
</thead>
<tbody>
<tr>
<td>PO</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>XLIFF</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>TBX</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Qt Linguist(.ts)</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Qt Phrasebook(.qph)</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>FOSS Tool</td>
<td>Virtaal</td>
<td>Pootle</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>Shortcuts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ctrl+Q</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+P</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Alt+Enter</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+Up</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Ctrl+Down</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Ctrl+PgUp</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+PgDown</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+F, F3</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+G</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Ctrl+Shift+G</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Alt+Left</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Alt+Right</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Alt+Down</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Shift+Enter</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Ctrl+Enter</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Ctrl+Shift+Enter</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>FOSS Tool</td>
<td>Virtaal</td>
<td>Pootle</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>Ctrl+Z</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Ctrl+1</td>
<td>✔️</td>
<td>❌</td>
</tr>
<tr>
<td>Ctrl+T</td>
<td>✔️</td>
<td>❌</td>
</tr>
</tbody>
</table>

### Navigation Modes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Virtaal</th>
<th>Pootle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incomplete Mode</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Workflow Mode</td>
<td>✔️</td>
<td>❌</td>
</tr>
<tr>
<td>Quality Check Mode</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Search Mode</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Translated</td>
<td>❌</td>
<td>✔️</td>
</tr>
<tr>
<td>Untranslated</td>
<td>❌</td>
<td>✔️</td>
</tr>
<tr>
<td>Suggestions</td>
<td>❌</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Table 7: Virtaal and Pootle Check-listing of features.

For what concerns the first section, Virtaal and Pootle provide more or less the same functionalities and few differences can be observed: on one hand, Pootle does not provide auto-completion and auto-correction, which would rather be useful to increase localisers’ fastness, on the other hand Virtaal lacks alternative source language and comment option. These latter are two important features: alternative source language is very helpful for localisers, since, as mentioned above (section 5.5.2.), they could better understand some passages/terms of the source text by having them in a “second” source language; while comments provide different contributors working on the same file with a direct means of communication in order to clarify some technical aspects or translation choices. Another
difference concerns placeables, as a matter of fact while in Virtaal they can be moved by means of shortcuts or simply clicking on them, on the other hand, Pootle only provides the click option. The second section of Table 7 shows that Virtaal and Pootle support the same formats. In the third section, we can notice that Virtaal provides localisers with more shortcuts than Pootle. This latter, in fact, only present some shortcuts for navigation (Ctrl+Up, Ctrl+Down) and actions within strings (Shift+Enter, Ctrl+Enter, Ctrl+Z). Although the variety of shortcuts provided by Virtaal could be an advantage especially for those who are used to work with the keyboard, once beginners learn the most common combinations, their efficiency is highly increased. Finally, the section comparing navigation modes shows that unlike Pootle, Virtaal does not allow localisers to move within translated/untranslated strings and suggestions. But, on the other hand, Pootle does not provide a workflow mode, which is present in Virtaal instead.

5.6. Conclusion

This chapter was an introduction to the evaluation process. We have briefly defined the purpose of the study (section 5.2). Then, we have analysed the users’ profile (section 5.3). We have also described all the materials involved in the study (section 5.4) which include: documents for the evaluation, NoteFly (section 5.4.1) – i.e. the application localised during the experiments – BB FlashBack Express (section 5.4.2), which was used to record and examine the participants’ tasks. In sections 5.5.1. and 5.5.2., we have analysed the localisation tools at the centre of this study: Pootle and Virtaal. In particular, we have detailed and compared their features by means of a checklist (section 5.5.3).
6. Evaluation process

6.1. Introduction

This chapter is based on the description of the evaluation process. At this stage, the scenario test described in section 4.4. was executed: a sort of real life environment was reproduced, where participants were asked to localise a little application, while being recorded on screen.

In section 6.2., we will analyse tests with Virtaal: we will describe the pilot study (section 6.2.1.) – which was performed at first in order to guarantee a correct execution of the experiments, preventing technical problems or misunderstandings – and then we will focus on the evaluation implementation (section 6.2.2.) explaining how the experiments took place. The same structure will be used to show tests with Pootle (section 6.3.): we will firstly describe the pilot study (section 6.3.1.) and then we will analyse the experiments execution (section 6.3.2.).

6.2. Tests with Virtaal

All the experiments conducted with Virtaal 0.7.1 were performed in the library of the FTI at the University of Geneva from 16th to 21st July 2014 by one participant at a time, four in total. Their involvement in the experiments did not exceed 1 hour. Tests were all executed using the same computer, i.e. a Toshiba laptop with the following specifications:

- **Processor:** Intel® Core™2 CPU T5200 @1.60 GHz 1.60 GHz
- **RAM:** 2.00 GB
- **Operating System:** Windows Vista™ Home Premium 32-bit

Each participant was provided with a folder on the desktop named as `vUSER1` for the first experiment, `vUSER2` for the second one, and so on. This folder contained: a PDF file providing some basic principles of software localisation, Virtaal instructions (also provided in a printed version), six PO files named with different language codes, including the `it.po` file to be localised, two DOC files consisting of a general questionnaire to be filled in before the tests (also provided in a printed version) and a final questionnaire to be
completed on screen at the end of the experiments. It is important to specify that, for both questionnaires, participants were free to leave questions unanswered. Before starting the experiments execution, all the documentation listed above, except from the PO file, was corrected by an English native speaker at first, and then by the supervisor of this thesis in order to guarantee the use of proper English grammar and syntax, and to avoid any wrong information or error which could cause misunderstandings.

As seen in section 5.4.1., Virtaal has local and external translation memories. In the context of these tests, we decided to only activate the local one, which included current translated files, in order to let participants test one of the most important Virtaal’s features without the risk of compromising the experiments. The use of external translation memories implies, in fact, that all the words/sentences, which have previously been translated and consequently “registered” on the server, are displayed to help localisers during their work. In this way, when participants to our experiments would have started their localisation task, they could have received innumerable matches from previous translations stored in the external translation memory. On the contrary, by using a local translation memory, we succeeded in preventing this problem. In particular, we “created” our translation memory by making a copy of the it.po file and translating only few complex sentences, including difficult and technical words, to facilitate participants’ work, since they had no knowledge of software localisation, apart from the brief introduction provided before starting the experiments. However, as translated strings were stored in a local memory, we were forced to delete the TM file, called tm.db, from the computer at the end of each experiment, in order not to compromise the following tests.

6.2.1. Pilot Study
The pilot study with Virtaal was performed on the 16th July 2014. Its main purpose was to find out the problems that could eventually be encountered by participants during the experiments. Therefore, it was executed by someone who had the same profile as the participants to the evaluation in order to obtain realistic results. In particular, thanks to the pilot study, we could identify and solve the following issues:

- The evaluation execution took too much time. The pilot user had to localise a small application of 1028 words, but not entirely, as a matter of fact 630 words had
already been translated, 389 were untranslated and 9 needed work. But since the pilot test lasted 1:55:53, we decided to translate some strings so that 855 words were already translated and participants only had to deal with 164 untranslated words and 9 needing work.

- Some aspects of the instructions led to confusion. We decided to separate information from commands by dividing the document into two parts: the first one consisting of a list of some useful tips and the second one providing participants with step-by-step instructions to execute their task. Then, some obscure passages were modified, in particular we added precise indications about the icons location in the user interface and we also included figures illustrating the cursor clicking on them.

- A question in the final questionnaire was not objective. As mentioned before (section 5.3.), questions must not influence participants’ answers. Therefore, we modified the subjective expression Fast Navigation in the neutral Navigation.

6.2.2. Evaluation Implementation

Before starting the experiments, participants were asked to read the information sheet, including specifications about the purpose, duration and confidentiality terms of the study, and to sign a consent form. Then, we showed and explained the content of a PDF file, providing a short introduction to software localisation and some FOSS basic principles, since participants to this study were non-experienced translators who did not know anything about localisation. Although everyone was free to ask for further clarifications on the subject, no question was formulated, probably because all notions were presented in a clear way, resulting from the documentation provided during classes at the Department of Translation Technology. Participants then completed a general questionnaire. Both screen and printed versions were available, but three participants out of four preferred the first one. At that stage, we provided participants with instructions, at which they could have a look before starting their task. Once we made sure that all was clear and participants were ready for the experiments, we activated the full-screen recording mode in BB FlashBack Express. In this way, we could record and subsequently analyse participants’ actions both on screen and on the keyboard. The evaluation started with a quick look at NoteFly, as a matter of fact, even if instructions suggested opening it from the desktop while translating
in order to figure out the context of some words/sentences, all participants explored the application before performing any other action, probably to have an idea of what they were going to localise. Afterwards, Virtaal was opened from the desktop in order to start the evaluation and its dashboard was displayed with access to helpful links and recent files. At that point, participants were asked to change preferences by inserting their user name and language team, i.e. Italian Team, in the specific fields. Then, they opened the it.po file in their folder directly from Virtaal window. Before performing their translation task, participants verified that the selected language combination was English » Italian and they activated the Incomplete mode in order to move between untranslated and fuzzy strings thus avoiding the already translated lines. Participants started their localisation with Virtaal, during which they were asked to revise the 6th target string, marked as Needs work, since the quality check box indicated that a placeable was missing, as shown in the figure below.

Fig. 8: Virtaal Quality Check box.

Once the translation was completed, participants saved their work and quitted Virtaal. At this stage, we stopped the recording mode in BB FlashBack Express in order to let participants complete the final questionnaire on screen.
6.3. Tests with Pootle

As for Virtaal, experiments with Pootle 2.5.1 were performed in the library of the FTI at the University of Geneva by one participant at a time, four in total. The experiments took place from 20\textsuperscript{th} November to 1\textsuperscript{st} December 2014 and participants’ involvement did not exceed 1 hour. Tests were executed on the same laptop used for Virtaal in order to avoid variables, which could invalidate the evaluation.

We contacted the administrators of the Sympa Pootle server\textsuperscript{26} by email to ask for their help in order to create a project on their server, including five NoteFly PO files. These latter actually contained the same text, but we renamed them for each participant to the evaluation and the pilot user. We had a doubt concerning Pootle projects: once a translation was submitted, was it stored in an internal, general translation memory? That might be a problem for the experiments, since when participants started their localisation task, they could receive the matches from the previous translations. In our exchanges with one of the administrators, we asked for further information about Pootle translation memory in order to know if it was possible to restrict it, as we did for experiments with Virtaal, by only enabling the local translation memory and deleting \textit{tm.db} file after each experiment. Following the solution suggested by the administrator, we decided to create 5 separate projects (one for each test) in order to avoid the problem caused by Pootle’s internal, general translation memory. We sent the five PO files, together with their French versions (\textit{fr.po}) to the administrator, because, even if NoteFly was localised from English to Italian, we wanted to provide participants with an Alternative source language to show them this Pootle functionality. The administrator created five translation projects for Notefly, where he added the above-mentioned files. They were available at the following links:

- \url{http://pootle.sympa.org/fr/notefly1/}
- \url{http://pootle.sympa.org/fr/notefly2/}

\textsuperscript{26} Sympa was developed in 1995. It is a customizable mailing list management tool which allows users to automate some laborious and time-consuming functions, such as subscriptions, list maintenance, archive and shared document management (Verdin et al., 2008).
In order to provide participants with access to these projects on the Sympa Pootle server, we created five Gmail accounts and passwords following the structure below, where the numbers changed in relation to the participant:

Email: pootle.user1@gmail.com  
Password: FOSSL10N1

Then, five Pootle accounts were created and activated from Gmail, so that participants could directly log in by inserting their username, which had the same structure as in tests with Virtaal in order to guarantee anonymity – that is to say pUSER1 for the first participant, pUSER2 for the second one and so on –, and password.

For the evaluation process, each participant was provided with a folder on the desktop named using the same logic as for tests with Virtaal, i.e. pUSER1 for the first experiment, pUSER2 for the second one, etc. This folder contained all the necessary documentation for the experiment execution: a PDF file providing some basic principles of software localisation, Pootle instructions (also provided in a printed version), a TXT file including participants’ username and password, two DOC files consisting of a general questionnaire (also provided in a printed version) and a final questionnaire to be completed on screen at the end of the experiments. As for experiments with Virtaal, participants were free to leave questions unanswered in both questionnaires.

6.3.1. Pilot Study

The pilot study with Pootle was executed on the 20th November 2014 by a participant with the same profile as the evaluators. Its main objective was the same as for Virtaal – that is to say identifying possible problems or misunderstandings to be solved before the experiments execution –, but in this case, since Pootle is a web-based tool, variables were more likely to occur, therefore we decided to perform the pilot study focusing mainly on the technical issues. As mentioned above (section 6.3.), we had created a Gmail account
(pootle.pilot.user@gmail.com for this test), with password (FOSSL10N) and then a Pootle account. Therefore, the participant logged in with his username (in this case, PILOTUSER) and password to access the Sympa Pootle server. He then clicked on the project named NoteFly to start his translation task.

The duration of the pilot study was not a problem – it lasted about 45 minutes –, since the participant only had to deal with 164 untranslated words and 9 needing work, resulting from the modifications introduced during the pilot study with Virtaal. The general questionnaire was the same as for Virtaal, while the final questionnaire had been modified: in particular we had changed some specific questions in relation to the executed task. The only problem with documentation was that a sentence was missing in the instructions, since an error had occurred while converting the DOC file format into Instructions.pdf.

For what concerns the technical part of this pilot study, we encountered the following issues:

- Alternative source language (French) was not displayed during the task. We initially thought it was due to the limited number of translated words in the fr.po file, so we added some translated strings. Since the problem was not solved, we opened other Sympa projects to verify if it was a general bug, but alternative source language was displayed, as shown in the figure below.

Fig. 9: Alternative source language in the Sympa user interface project.
Since we did not succeed in finding a solution to this bug, we decided to eliminate questions about Alternative source language. We only left a reference in the instructions, because we would like evaluators to know that Pootle normally provides this functionality.

- Placeables were not recognised by the system, therefore the participant had to insert them manually in the translated strings instead of just clicking on them. In order to find out the source of this problem, we checked the other projects in the server, where we found that placeables were recognized. Afterwards, we noticed that this bug seemed to be specific to some code elements, such as variables and numbers, which were treated like normal text in our projects, while some others, like tags, for example, were perfectly recognized. For this reason, we decided to modify the PO file to be translated in the evaluation: we eliminated the translation provided for some sentences including placeables recognised by the system. In this way evaluators had to translate 194 words instead of 164 (Virtaal's evaluation), but, since Pootle’s translation memory facilitated participants localisation task by providing more matches than Virtaal, the two evaluations were balanced.

- Downloading the translated file was not possible. We checked the other Sympa projects, but every time we clicked on Download (zip), the following message was displayed: ‘The webpage cannot be found’. We therefore realised that it was a bug concerning all the projects present in the server. Nonetheless, we decided to leave instructions unchanged, because, since reliability was not among the characteristics and sub-characteristics to be tested, this problem did not influence our evaluation.

6.3.2. Evaluation Implementation

The evaluation process was preceded by a sort of introductory stage: evaluators read the information sheet, signed a consent form and they were provided with a short introduction to software localisation and some FOSS basic principles. All participants then completed a general questionnaire on screen, although it was also available in a printed version. Then, instructions were provided to evaluators, who quickly read them in order to have an idea of the task they were going to perform. During the experiments, all participants used screen instructions, probably because in this way they only needed to click on links instead of
typing full URLs from the printed version. To start the evaluation, we activated the full-screen recording mode in BB FlashBack Express, which allowed us to record participants’ actions on screen in order to analyse them. After having a look at NoteFly, evaluators clicked on the link http://pootle.sympa.org/ to access Pootle directly from the instructions. Then, they logged in by copying user name and password from the .txt file in their folder and selecting English as the Interface Language. Once logged in, participants modified their language preferences selecting French as alternative source language. They saved changes and went back to the main page represented in the figure below.

![Sympa Pootle server main page.](image)

Fig. 10: Sympa Pootle server main page.

At this stage, evaluators selected their project from the shown list and then clicked on their target language (Italian) and on Translate to access the translation interface. As for the evaluation with Virtaal, since some strings had already been translated, participants activated the Incomplete mode and then started localising NoteFly. They had to revise the 6th string, marked as Needs work, because a placeable was missing in the Italian translation. They corrected the error and added a comment to explain what they had just done and why. Once the translation finished, evaluators were asked to download their work, by clicking on the link, and then save the .po file in their folder on the desktop. But as explained before (section 6.3.1.), it was not possible to download the translated files. Therefore, they had to quit Pootle without having a copy of their work. At this stage, we
disabled the recording mode in BB FlashBack Express, so that evaluators could complete the final questionnaire on screen.

6.4. Conclusion

This chapter was based on the evaluation process and, in particular, on the implementation of the scenario test. In section 6.2., we focused on tests with Virtaal: we showed the results of the pilot study by analysing encountered issues and applied solutions (section 6.2.1.) and we described the evaluation implementation (section 6.2.2.). Tests with Pootle were at the centre of section 6.3.: we firstly focused on the pilot study (section 6.3.1.), which was especially useful for the experiments with Pootle, and then we analysed the evaluation execution (section 6.3.2.).
7. Results

7.1. Introduction

By following the EAGLES 7-step recipe, we described the evaluation execution in the previous chapter and we are now going to analyse the collected data by applying the predefined metrics (section 4.3.). These results from both objective (section 7.2.) and subjective (section 7.3.) approaches will be at the centre of the chapter. First of all, we will focus on videos analysis, from which it was possible to evaluate the tools’ time behaviour (section 7.2.1.) and operability (section 7.2.2.). Then, questionnaires results will be compared to determine the tools’ suitability (section 7.3.1.), accuracy (section 7.3.2.) efficiency (section 7.3.3.), understandability (section 7.3.4.), learnability (section 7.3.5.) and attractiveness (section 7.3.6.). In section 7.4., we will provide a summary of the obtained results in order to make the comparison clearer. Finally, in section 7.5. we will discuss the obtained results in relation to the purpose of our evaluation.

7.2. Objective Approach

As previously discussed (section 4.5), two different approaches were applied in this study. For what concerns the objective evaluation, during the experiments, participants’ actions on screen and on the keyboard were video recorded by the means of BB FlashBack Express. In order to evaluate Virtaal and Pootle’s efficiency and operability, we are now going to analyse the results obtained for the time behaviour and the number of clicks, keystrokes and open windows performed during the experiments.

7.2.1. Time Behaviour

In order to evaluate Virtaal and Pootle’s efficiency, we are going to analyse results of their time behaviour during the scenario test recorded with BB FlashBack Express, by dividing each experiment into 3 parts: **introduction**, determining participants’ familiarisation time, **translation task**, delineating the time taken to translate part of NoteFly GUI, **revision of the 4th string**, measuring the revision time of a sentence where a placeable was missing by following the provided instructions. From a general analysis of results, we can affirm that participants using Virtaal took less time for all the evaluation stages, as shown in the tables below.
Table 8: Time behaviour analysis for Virtaal.

<table>
<thead>
<tr>
<th>User</th>
<th>Introduction</th>
<th>Translation Task</th>
<th>Revision 4th string</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>2m95s</td>
<td>39m19s</td>
<td>0m27s</td>
<td>43m45s</td>
</tr>
<tr>
<td>V2</td>
<td>5m03s</td>
<td>11m03s</td>
<td>0s</td>
<td>17m13s</td>
</tr>
<tr>
<td>V3</td>
<td>7m28s</td>
<td>15m21s</td>
<td>0m37s</td>
<td>21m20s</td>
</tr>
<tr>
<td>V4</td>
<td>6m25s</td>
<td>23m08s</td>
<td>0m50s</td>
<td>29m16s</td>
</tr>
<tr>
<td>Average</td>
<td>5m37s</td>
<td>22m12s</td>
<td>0m27s</td>
<td>27m73s</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>1m86s</td>
<td>12m42s</td>
<td>0m28s</td>
<td>11m60s</td>
</tr>
</tbody>
</table>

Table 9: Time behaviour analysis for Pootle.

<table>
<thead>
<tr>
<th>User</th>
<th>Introduction</th>
<th>Translation Task</th>
<th>Revision 4th string</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>6m32s</td>
<td>1h02m40s</td>
<td>1m23s</td>
<td>1h16m22s</td>
</tr>
<tr>
<td>P2</td>
<td>6m36s</td>
<td>25m03s</td>
<td>0m21s</td>
<td>32m25s</td>
</tr>
<tr>
<td>P3</td>
<td>4m35s</td>
<td>22m33s</td>
<td>0m34s</td>
<td>29m52s</td>
</tr>
<tr>
<td>P4</td>
<td>4m50s</td>
<td>36m09s</td>
<td>0m39s</td>
<td>42m54s</td>
</tr>
<tr>
<td>Average</td>
<td>5m38s</td>
<td>46m46s</td>
<td>0m54s</td>
<td>55m13s</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>1m10s</td>
<td>37m76s</td>
<td>0m46s</td>
<td>41m10s</td>
</tr>
</tbody>
</table>

The data collected in Tables 8 and 9 indicate that Virtaal and Pootle’s familiarisation time was nearly the same, respectively $\bar{x} = 5m37s$ ($sd = 1.86$) and $\bar{x} = 5m38s$ ($sd = 1.10$).

The performed translation task was shorter in tests with Virtaal ($\bar{x} = 22m12s$, $sd = 12.42$) than in Pootle’s experiments ($\bar{x} = 46m46s$, $sd = 37.76$). In this case, it must however be underlined that User P1 had to deal with an Internet connection problem, therefore the webpage had to be refreshed causing strings disorder and the participant had to retranslate some sentences which had already been translated. Nonetheless, if we calculate the total time behaviour without considering this user, tests with Pootle were still longer ($\bar{x} = 34m77s$, $sd = 6.68$).

During the revision stage, Virtaal’s users took less time ($\bar{x} = 0m27s$, $sd = 0.28$) than Pootle’s evaluators ($\bar{x} = 0m54s$, $sd = 0.46$) to correct the 4th string by adding the missing placeable. However, videos showed that User V2 completely ignored placeables during the whole experiment.

Finally, the tables above show that all experiments with Virtaal were shorter ($\bar{x} = 27m73s$, $sd = 11.60$) than Pootle’s tests ($\bar{x} = 55m13s$, $sd = 41.10$).
7.2.2. Operability

Virtaal and Pootle’s operability was evaluated by analysing participants’ videos in order to count mouse clicks, keystrokes and open windows throughout the experiments.

The analysis of mouse clicks was performed for three particular phases: preferences change, including for example the selection of the language pair, navigation within strings and placeables manipulation. For Virtaal, the total clicks were calculated from the moment participants opened the translation file to the last click performed quitting the application. For Pootle, we considered clicks from participants’ log in to their last click to quit the tool. Overall, Virtaal evaluators needed far fewer clicks to execute actions, that is to say $\bar{x} = 46.25$ clicks ($sd = 37.58$) against Pootle’s $\bar{x} = 209.75$ clicks ($sd = 138.38$).

<table>
<thead>
<tr>
<th>User</th>
<th>Total clicks</th>
<th>Preferences Change</th>
<th>Navigation within strings</th>
<th>Placeables Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>101</td>
<td>4</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>V2</td>
<td>17</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V3</td>
<td>39</td>
<td>4</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>V4</td>
<td>28</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Average</td>
<td>46.25</td>
<td>4.25</td>
<td>2.25</td>
<td>2.50</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>37.58</td>
<td>0.50</td>
<td>4.50</td>
<td>3.31</td>
</tr>
</tbody>
</table>

Table 10: Number of clicks for Virtaal.

<table>
<thead>
<tr>
<th>User</th>
<th>Total clicks</th>
<th>Preferences Change</th>
<th>Navigation within strings</th>
<th>Placeables Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>211</td>
<td>12</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>P2</td>
<td>108</td>
<td>37</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>P3</td>
<td>115</td>
<td>13</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>P4</td>
<td>405</td>
<td>10</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>209.75</td>
<td>18</td>
<td>1</td>
<td>3.75</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>138.38</td>
<td>12.72</td>
<td>1.15</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Table 11: Number of clicks for Pootle.

As shown in the tables above, in order to change their preferences, users testing Virtaal executed a lesser amount of clicks ($\bar{x} = 4.25$ clicks, $sd = 0.50$) than Pootle’s evaluators ($\bar{x} = 18$ clicks, $sd = 12.72$).
For what concerns navigation within strings, tests with Virtaal surprisingly registered a larger number of clicks ($\bar{x} = 2.25$ clicks, $sd = 4.50$) than Pootle’s experiments ($\bar{x} = 1$ click, $sd = 1.15$). However, we later discovered that these results were influenced by an outlier (User V1). In order to determine the reason of this result, we further analysed the video of this user, who was the only one performing some clicks, precisely 9. From this analysis we noticed that the participant did not initially use the keyboard to move within strings, because he still had to familiarise with the tool, therefore for the first 9 strings, he clicked on the translation status bar in order to change it into Translated.

Placeables manipulation with Virtaal was characterised by an average of 2.50 clicks ($sd = 3.31$), while Pootle’s evaluators executed an average of 3.75 clicks ($sd = 0.95$). The difference is quite clear, but it must be specified that User P2 executed 5 clicks because he tried to copy and paste placeables instead of simply clicking on them so that they could directly be added to the translated string. In addition, as mentioned above, User V2 did not move any placeables and this could have somehow influenced results. However, thanks to our previous Feature Inspection, we know that, contrary to what happens with Pootle, placeables manipulation with Virtaal is also possible by the means of shortcuts, thus allowing users to directly insert them in their translation without clicking.

The analysis of keystrokes was carried out using BB Flashback Express, which generated an XML file, containing all keys pressed by participants divided into normal keys, including letters, numbers and characters, and virtual keys$^{27}$, consisting of commands and shortcuts. These latter were at the centre of our analysis and they were divided into two separate lists for both Virtaal and Pootle.

---

$^{27}$ We used here BB Flashback terminology.
<table>
<thead>
<tr>
<th>VIRTUAL KEYS</th>
<th>VICTAAL</th>
<th>POOTLE</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commands</td>
<td>ArwDwn</td>
<td>ArwDwn</td>
<td>Move down</td>
</tr>
<tr>
<td></td>
<td>ArwLft</td>
<td>ArwLft</td>
<td>Move left</td>
</tr>
<tr>
<td></td>
<td>ArwRght</td>
<td>ArwRght</td>
<td>Move right</td>
</tr>
<tr>
<td></td>
<td>ArwUp</td>
<td>ArwUp</td>
<td>Move up</td>
</tr>
<tr>
<td></td>
<td>CpLk</td>
<td>CpLk</td>
<td>Enable/Disable uppercase</td>
</tr>
<tr>
<td></td>
<td>Del</td>
<td>Del</td>
<td>Delete</td>
</tr>
<tr>
<td></td>
<td>PgUp</td>
<td>PgUp</td>
<td>Scroll the page up</td>
</tr>
<tr>
<td></td>
<td>Ret</td>
<td>Ret</td>
<td>Move to the next string</td>
</tr>
<tr>
<td></td>
<td>TAB</td>
<td>TAB</td>
<td>Accept auto-completion suggestions</td>
</tr>
<tr>
<td>Shortcuts</td>
<td>[Ctrl]+[8]</td>
<td>[Ctrl]+[8]</td>
<td>Accept TM suggestion</td>
</tr>
<tr>
<td></td>
<td>[Alt]+[ArwDwn]</td>
<td>[Alt]+[ArwDwn]</td>
<td>Copy text from source to target</td>
</tr>
<tr>
<td></td>
<td>[Alt]+[ArwLft]</td>
<td>[Alt]+[ArwLft]</td>
<td>Insert placeables in the target string</td>
</tr>
<tr>
<td></td>
<td>[Alt]+[ArwRght]</td>
<td>[Ctrl]+[1]</td>
<td>Select the previous placeable</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[Alt]+[ArwDwn]</td>
<td>[Ctrl]+[Alt]+[ArwDwn]</td>
<td>Select the next placeable Use the first translation suggestion</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[ArwDwn]</td>
<td>[Ctrl]+[ArwDwn]</td>
<td>No function</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[ArwLft]</td>
<td>[Ctrl]+[ArwLft]</td>
<td>Move to the next string</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[ArwRght]</td>
<td>[Ctrl]+[ArwRght]</td>
<td>Move to the beginning of a word</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[ArwUp]</td>
<td>[Ctrl]+[ArwUp]</td>
<td>Move to the previous string</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[C]</td>
<td>[Ctrl]+[C]</td>
<td>Copy</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[Ret]</td>
<td>[Ctrl]+[Ret]</td>
<td>Move to the next string</td>
</tr>
<tr>
<td></td>
<td>[Ctrl]+[V]</td>
<td>[Ctrl]+[V]</td>
<td>Paste</td>
</tr>
<tr>
<td></td>
<td>[Shift]+[ArwLft]</td>
<td>[Shift]+[ArwLft]</td>
<td>Select text</td>
</tr>
<tr>
<td></td>
<td>[Shift]+[Ret]</td>
<td>[Shift]+[Ret]</td>
<td>Start new line</td>
</tr>
</tbody>
</table>

Table 12: Lists of Virtual Keys for Virtaal and Pootle.

The table above shows that for both Virtaal and Pootle participants used seven types of commands. These latter are simple and common, since they are well known to any computer user. For what concerns shortcuts, 11 different combinations were used during experiments with Virtaal, a little bit more than in tests with Pootle, where participants used

28 The user in question was here trying to insert a placeable in the target string by using the wrong shortcut.
eight types. This can be explained by the results of our previous Feature Inspection, where we observed that Virtaal provides more shortcuts than Pootle, which only present few shortcuts for navigation and actions within strings. Apart from Ctrl+C and Ctrl+V, shortcuts used in this study were not simple, since they were provided to participants by some specific instructions. The table below shows the number of Keys and Virtual Keys for Virtaal and Pootle.

<table>
<thead>
<tr>
<th>USER</th>
<th>Keys</th>
<th>Virtual Keys</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>225</td>
<td>37</td>
<td>262</td>
</tr>
<tr>
<td>V2</td>
<td>195</td>
<td>5</td>
<td>200</td>
</tr>
<tr>
<td>V3</td>
<td>106</td>
<td>21</td>
<td>127</td>
</tr>
<tr>
<td>V4</td>
<td>110</td>
<td>13</td>
<td>123</td>
</tr>
<tr>
<td>Average</td>
<td>159</td>
<td>19</td>
<td>178</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>60.17</td>
<td>13.66</td>
<td>66.24</td>
</tr>
</tbody>
</table>

Table 13: Number of keystrokes for Virtaal.

<table>
<thead>
<tr>
<th>USER</th>
<th>Keys</th>
<th>Virtual Keys</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>329</td>
<td>16</td>
<td>345</td>
</tr>
<tr>
<td>P2</td>
<td>122</td>
<td>54</td>
<td>176</td>
</tr>
<tr>
<td>P3</td>
<td>181</td>
<td>56</td>
<td>237</td>
</tr>
<tr>
<td>P4</td>
<td>320</td>
<td>54</td>
<td>374</td>
</tr>
<tr>
<td>Average</td>
<td>238</td>
<td>45</td>
<td>283</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>102.81</td>
<td>19.35</td>
<td>92.53</td>
</tr>
</tbody>
</table>

Table 14: Number of keystrokes for Pootle.

As described in Table 13, participants to experiments with Pootle pressed more keys ($\bar{x} = 283$ keys, $sd = 92.53$) than Virtaal’s evaluators ($\bar{x} = 178$, $sd = 66.24$). For what concerns virtual keys, results are the same as for normal keys, that is to say that commands and shortcuts performed during tests with Pootle ($\bar{x} = 45$ virtual keys, $sd = 19.35$) are more than those pressed by Virtaal's users ($\bar{x} = 19$ virtual keys, $sd = 13.66$). Overall, we could estimate that the total number of keystrokes was shorter for the experiments with Virtaal.

The last aspect to be considered in the evaluation of operability is the analysis of the number of windows opened during the tests. The tables below show that, during the experiments, Pootle needed more open windows ($\bar{x} = 3.50$, $sd = 1$) than Virtaal ($\bar{x} = 2.50$, $sd = 1$).
Nonetheless, we estimated that the number of open windows for the experiments with Virtaal and Pootle was the same ($\bar{x} = 2$, $sd = 0$), by considering two fundamental aspects:
1. During experiments with Pootle, all participants preferred following instructions on screen, while for Virtaal all evaluators used instructions in printed version.

2. Wikipedia and Google windows were opened to look for some terms from the source text, therefore they could not be taken into consideration, since this depends on participants’ comprehension of the source texts rather than on tools themselves.

We can thus affirm that for both tools, evaluators had to open two windows: NoteFly, to have a clearer idea of the application, and Virtaal or Pootle, depending on the experiments in question.

7.3. Subjective Approach

After having described the data resulting from the objective approach (section 7.2.), we are now going to analyse the results obtained from questionnaires, representing participants’ subjective evaluation. In order to evaluate Virtaal and Pootle’s functionality and usability we are going to examine results for suitability and accuracy on one hand, and for understandability, learnability and attractiveness on the other hand. Results for efficiency will also be observed and described.

7.3.1. Suitability

Virtaal and Pootle’s suitability was evaluated through participants’ opinions on the interface and features of the tools. Six questions were asked for Virtaal (VQ3, VQ4, VQ5, VQ8, VQ9, VQ10) and four for Pootle (PQ3, PQ4, PQ5, PQ8).

In VQ3 and PQ3, we asked participants whether the icons and menus in the tools looked like some others they had already used. On a scale of 1 to 6, where 1 indicated Completely disagree and 6 Completely agree, participants rated Virtaal better (\(\bar{x} = 4, \text{sd} = 2\)), than Pootle (\(\bar{x} = 2.50, \text{sd} = 0.57\)). But none of the participants who agreed to this question provided us with more information when we asked them to specify the tools they had thought of (VQ4, PQ4).

In VQ5 and PQ5, we asked participants to give a score from 1 (very poor) to 6 (excellent) to a list of features of Virtaal and Pootle GUI. As shown in Table 17, overall Pootle obtained a higher score for the following variables: layout (Pootle \(\bar{x} = 4.25, \text{sd} = 0.50\); Virtaal \(\bar{x} = 3.25, \text{sd} = 0.95\)), clarity (Pootle \(\bar{x} = 4.50, \text{sd} = 0.57\); Virtaal \(\bar{x} = 4.25, \text{sd} =

86
0.50), intuitiveness (Pootle $\bar{x} = 4.50$, $sd = 0.57$; Virtaal $\bar{x} = 4$, $sd = 1.15$), familiarity (Pootle $\bar{x} = 3.50$, $sd = 1$; Virtaal $\bar{x} = 3.25$, $sd = 0.50$) and design (Pootle $\bar{x} = 3.25$, $sd = 1.29$; Virtaal $\bar{x} = 2.50$, $sd = 0.95$). Responsiveness was the only feature for which Virtaal had a better result ($\bar{x} = 4.50$, $sd = 0.57$) than Pootle ($\bar{x} = 3.25$, $sd = 0.50$).

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>VIRTAAAL</th>
<th>POOTLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layout</td>
<td>$\bar{x} = 3.25$, $sd = 0.95$</td>
<td>$\bar{x} = 4.25$, $sd = 0.50$</td>
</tr>
<tr>
<td>Clarity</td>
<td>$\bar{x} = 4.25$, $sd = 0.50$</td>
<td>$\bar{x} = 4.50$, $sd = 0.57$</td>
</tr>
<tr>
<td>Intuitiveness</td>
<td>$\bar{x} = 4$, $sd = 1.15$</td>
<td>$\bar{x} = 4.50$, $sd = 0.57$</td>
</tr>
<tr>
<td>Familiarity</td>
<td>$\bar{x} = 3.25$, $sd = 0.50$</td>
<td>$\bar{x} = 3.50$, $sd = 1$</td>
</tr>
<tr>
<td>Responsiveness</td>
<td>$\bar{x} = 4.50$, $sd = 0.57$</td>
<td>$\bar{x} = 3.25$, $sd = 0.50$</td>
</tr>
<tr>
<td>Design</td>
<td>$\bar{x} = 2.50$, $sd = 1.29$</td>
<td>$\bar{x} = 3.25$, $sd = 0.95$</td>
</tr>
</tbody>
</table>

Table 17: Summary of participants’ answers to VQ5 and PQ5.

VQ8 and VQ9 were specifically formulated for experiments with Virtaal. Answers to these close-ended questions were provided on the basis of a scale, where Completely disagree was scored as 1, and Completely agree as 6. In VQ8, we asked if Virtaal’s suggestions while typing were annoying and answers were quite different: two participants slightly disagreed, another one disagreed and the last one slightly agreed. This question obtained an average score of 4 ($sd = 0.81$). For what concerns VQ9, we would like to know if the messages displayed in Virtaal Quality Check box were always clear for evaluators. Here again answers were different, but quite positive, with an average score of 4.75 ($sd = 0.95$),

For VQ8 we had to adapt the scale in relation to other scores, where the highest one represents a better result.
two participants out of four slightly agreed, another one agreed and the last one completely agreed.

In VQ10 and PQ8, on a scale of 1 to 6, where 1 indicated very poor and 6 excellent, participants had to give a score to Virtaal’s features in relation to their usefulness in the localisation process.

<table>
<thead>
<tr>
<th>FEATURES</th>
<th>VIRTAAL</th>
<th>POOTLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td>$\bar{x} = 4.50$</td>
<td>$\bar{x} = 4.50$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.29$</td>
<td>$sd = 0.57$</td>
</tr>
<tr>
<td>Translation memory</td>
<td>$\bar{x} = 5.25$</td>
<td>$\bar{x} = 3.50$</td>
</tr>
<tr>
<td></td>
<td>$sd = 0.95$</td>
<td>$sd = 0.57$</td>
</tr>
<tr>
<td>Quality check</td>
<td>$\bar{x} = 4.25$</td>
<td>$\bar{x} = 2.75$</td>
</tr>
<tr>
<td></td>
<td>$sd = 0.95$</td>
<td>$sd = 0.50$</td>
</tr>
<tr>
<td>Terminology help</td>
<td>$\bar{x} = 4$</td>
<td>$\bar{x} = 4.25$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.41$</td>
<td>$sd = 0.95$</td>
</tr>
<tr>
<td>Placeables manipulation</td>
<td>$\bar{x} = 4.50$</td>
<td>$\bar{x} = 4.25$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.29$</td>
<td>$sd = 0.50$</td>
</tr>
<tr>
<td>Web look-ups</td>
<td>$\bar{x} = 3.50$</td>
<td>$\bar{x} = 4.25$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.29$</td>
<td>$sd = 0.50$</td>
</tr>
</tbody>
</table>

Table 18: Summary of answers to VQ10 and PQ8.

Results shown in the table above indicate that navigation obtained the same score for both Virtaal ($\bar{x} = 4.50$, $sd = 1.29$) and Pootle ($\bar{x} = 4.50$, $sd = 0.57$). Participants rated Virtaal better than Pootle for what concerns translation memory (Virtaal $\bar{x} = 5.25$, $sd = 0.95$; Pootle $\bar{x} = 3.50$, $sd = 0.57$), quality check (Virtaal $\bar{x} = 4.25$, $sd = 0.95$; Pootle $\bar{x} = 2.75$, $sd = 0.50$) and placeables manipulation (Virtaal $\bar{x} = 4.50$, $sd = 1.29$; Pootle $\bar{x} = 4.25$, $sd = 0.50$). On the other hand, Pootle obtained a higher score for terminology help (Pootle $\bar{x} = 4.25$, $sd = 0.95$; Virtaal $\bar{x} = 4$, $sd = 1.41$) and web look-ups (Pootle $\bar{x} = 4.25$, $sd = 0.50$; Virtaal $\bar{x} = 3.50$, $sd = 1.29$).
7.3.2. Accuracy

Virtaal and Pootle’s accuracy was evaluated by the means of six questions for each of the two tools. Participants’ opinions were collected concerning Virtaal and Pootle’s performance in relation to the accomplished tasks. Questions VQ1, VQ2, VQ6, VQ7, VQ11 and VQ12 were taken into consideration for Virtaal, while PQ1, PQ2, PQ6, PQ7, PQ9 and PQ10 were analysed for Pootle.

Participants answered to VQ1 and PQ1 indicating to which extent they agreed with the statement saying that the tool in question did whatever they needed it to do during their translation task\(^{30}\). Virtaal obtained a higher score ($\bar{x} = 4.75$, $sd = 0.50$) than Pootle ($\bar{x} = 4$, $sd = 0$). Two Pootle evaluators, who slightly agreed, motivated their answer by specifying that they could not download their translated file once finished.

In VQ2 and PQ2, we asked if all needed information was present on the tools’ interface\(^{31}\). Evaluators rated Pootle better ($\bar{x} = 4.50$, $sd = 0.57$) than Virtaal ($\bar{x} = 4.25$, $sd = 1.50$), since three users agreed and one disagreed with VQ2, while two users agreed and two slightly agreed with PQ2.

Concerning placeables, participants answered to two questions\(^{32}\). In VQ6, all four Virtaal’s evaluators agreed on the fact that placeables could be easily identified ($\bar{x} = 5$, $sd = 0$). The only difference with Pootle’s answers ($\bar{x} = 4.75$, $sd = 0.50$) was that in this case a participant out of four slightly agreed with our statement, explaining that some placeables did not ‘seem to be recognized by Pootle’ (PQ6). In the second question about placeables, which asked if their manipulation reduces the risk of typos, we obtained the same result ($\bar{x} = 4.75$, $sd = 0.50$) for both Virtaal (VQ7) and Pootle (PQ7). As a matter of fact, in both tools’ questionnaires three participants out of four agreed and the other one slightly agreed.

In VQ11 and PQ9, we asked if all Virtaal and Pootle’s contents were visible and accessible\(^{33}\). Virtaal obtained a higher score ($\bar{x} = 4.50$, $sd = 0.57$) – two participants agreed and the other two slightly agreed with VQ11 – than Pootle ($\bar{x} = 4.25$, $sd = 0.50$). Only an

\(^{30}\) A scale of 1 (Completely disagree) to 6 (Completely agree) was used.

\(^{31}\) A scale of 1 (Completely disagree) to 6 (Completely agree) was used.

\(^{32}\) A scale of 1 (Completely disagree) to 6 (Completely agree) was used.

\(^{33}\) A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
evaluator agreed with PQ9, while the other three slightly agreed and wrote in the further comments field in order to justify their answers by underlining that downloading the translated files could not be possible at the end of their experiments.

In both VQ12 and PQ10, an evaluator agreed with the statement saying that the tool reliability was satisfying and the other three slightly agreed. Virtaal and Pootle thus obtained the same score ($\bar{x} = 4.25, sd = 0.50$).

7.3.3. Efficiency

Efficiency was evaluated analysing results of five questions for Virtaal (VQ13, VQ14, VQ15, VQ16, VQ17) and five for Pootle (PQ11, PQ12, PQ13, PQ14, PQ15).

In VQ13 and PQ11, we asked if the tool took too much time to get something done. For these close-ended questions, we had to adapt the scale by following the other scores, where 6 represented the best and 1 the worst. Therefore, here completely agree corresponded to 1 and completely disagree to 6. Virtaal and Pootle obtained the same score ($\bar{x} = 4.75, sd = 0.50$).

When we asked evaluators if the tool increased their efficiency$^{34}$ (VQ14 and PQ12), they rated Pootle better ($\bar{x} = 5.25, sd = 0.50$) than Virtaal ($\bar{x} = 4.50, sd = 1$).

In VQ15 and PQ13 we asked evaluators to indicate to what extent they agreed on the fact that the tool improved the localisation process$^{35}$. Virtaal obtained a higher score ($\bar{x} = 5, sd = 0.81$) than Pootle ($\bar{x} = 4.75, sd = 0.50$).

In VQ16 and PQ14$^{36}$, when participants were asked if working mostly with the keyboard was helpful in saving time, they rated Virtaal better ($\bar{x} = 5.25, sd = 0.50$) than Pootle ($\bar{x} = 3.75, sd = 0.95$). An evaluator who agreed with PQ14 suggested that ‘maybe for those who are not used to work with the keyboard it could have the opposite effect’. This is probably the reason of the inconsistency within Pootle’s evaluators’ answers.

---

$^{34}$ A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
$^{35}$ A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
$^{36}$ A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
In VQ17 and PQ15, we asked participants if the time needed to localise NoteFly was acceptable\textsuperscript{37}. Virtaal and Pootle obtained the same score ($\bar{x} = 5.25$, $sd = 0.50$), since for both tools three evaluators agreed and one completely agreed.

7.3.4. Understandability

Two questions were used to evaluate Virtaal and Pootle’s understandability: VQ19/PQ17 and VQ20/PQ18.

In VQ19 and PQ17, when participants were asked to indicate if given instructions were clear and adequate\textsuperscript{38}, Virtaal obtained a lower score ($\bar{x} = 4.50$, $sd = 0.57$) than Pootle ($\bar{x} = 5.50$, $sd = 0.57$).

In VQ20 and PQ18\textsuperscript{39}, we asked participants if they would have been able to use the tools only referring to its documentation, or online help for Virtaal. In this case, Virtaal obtained a better result ($\bar{x} = 4.75$, $sd = 0.95$) than Pootle ($\bar{x} = 4$, $sd = 0$).

7.3.5. Learnability

In order to evaluate the tools’ learnability, we asked two questions for both Virtaal (VQ18, VQ26) and Pootle (PQ16, PQ24).

In VQ18 and PQ16, participants were asked to evaluate the difficulty encountered during the experiments in learning how to use the tool, on a scale of 1 to 6, where the latter represented the highest level of difficulty. Here again we have to adapt the scale to other scores, by attributing 6 to the lowest level of difficulty (1) and 1 to the highest (6). Virtaal had a worse score ($\bar{x} = 4.75$, $sd = 0.50$) than Pootle ($\bar{x} = 5$, $sd = 0.81$).

In VQ26 and PQ24, we found out that all Virtaal and Pootle’s evaluators thought that the tools are appropriate for beginners in software localisation ($\bar{x} = 6$, $sd = 0$). One of the four Pootle’s users also commented his answer by specifying that he was a beginner and ‘didn’t encounter any difficulty’.

\textsuperscript{37} A scale of 1 (Completely disagree) to 6 (Completely agree) was used.

\textsuperscript{38} A scale of 1 (Completely disagree) to 6 (Completely agree) was used.

\textsuperscript{39} A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
7.3.6. Attractiveness

In order to evaluate attractiveness, indicating participants’ overall satisfaction, eight questions were formulated for both Virtaal (VQ21, VQ22, VQ23, VQ24, VQ25, VQ27, VQ28, VQ29) and Pootle (PQ19, PQ20, PQ21, PQ22, PQ23, PQ25, PQ26, PQ28), with the addition of an extra question in Pootle’s questionnaire (PQ27).

In VQ21 and PQ19 we asked participants if they were satisfied with the resulting localisation\textsuperscript{40}. Pootle obtained a better result ($\overline{x} = 5$, $sd = 0$) than Virtaal ($\overline{x} = 4.75$, $sd = 0.50$).

By answering to VQ22 and PQ20, all participants affirmed they enjoyed working with the tools ($\overline{x} = 6$, $sd = 0$).

In VQ23, evaluators had to indicate the best thing about working with Virtaal. All users affirmed, in different ways, that Virtaal was fast. One of the users also stated that it was ‘easy to learn’. For PQ21, Pootle’s evaluators gave quite different answers. User 1 wrote ‘Share knowledges [sic]’, probably referring to the fact that Pootle is a collaborative web-based system where innumerable users’ translations/competences merge. User 2 affirmed that Pootle was ‘easy and useful’. Then, User 3 wrote that TM and terminology help were useful to save time and User 4 indicated TM as the best thing about working with Pootle, ‘because it made the translation process quicker’.

In VQ24, we asked for the most annoying thing about working with Virtaal. Two Virtaal’s users affirmed that the fact that they could not see the strings in context was the most annoying thing. User 4 specified that because of this ‘it was difficult in some cases to think of the most appropriate equivalent in Italian’ Virtaal’s suggestions were also indicated as an annoying thing in two users’ answers. Concerning Pootle (PQ22), three users out of four answered: User 1 noticed some lacks in the translation memory, User 2 designated internet connection as the most annoying thing and User 4 was bothered by the fact that ‘there was no percentage matching in the sentences proposed by the translation memory’.

By answering to VQ25 and PQ23 all participants affirmed that both Virtaal and Pootle were good software localisation tools.

\textsuperscript{40} A scale of 1 (Completely disagree) to 6 (Completely agree) was used.
In **VQ27**, evaluators listed the main advantages of Virtaal. Almost all of them (three out of four) wrote that the tool was clear, intuitive, easy to use and rapid. A different answer was given by User 4, who affirmed that Virtaal reduced ‘*external interference to a minimum*’, since ‘*web search tool is incorporated*’ in it. For what concerns Pootle (**PQ25**), User 1 indicated terminology help as its main advantage, while User 2 affirmed that the tool had easy research modes and a very good interface. In addition, he underlined the fact that Pootle is ‘*always updated thanks to internet*’. Finally, Users 3 and 4 answered that Pootle was intuitive and easy to use.

By answering to **VQ28**, evaluators indicated Virtaal’s main disadvantages. Users 1 and 4 affirmed that they disliked layout, which was not very clear. User 4 also added the lack of context to his list of Virtaal’s disadvantages. Concerning Pootle, three users out of four answered to **PQ26**. User 1 noticed that ‘*the terms insertion flow is probably not enough fast*’, while User 3 underlined that ‘*sometimes translation memory doesn’t recognize some already translated sentences*’. Lastly, User 4 affirmed that the fact that everyone could translate/modify translations in Pootle ‘*is not always good for the translation quality*’.

An extra question was provided to Pootle’s evaluators in order to find out their opinion about working online, in particular they had to explain whether it was an advantage or disadvantage and why (**PQ27**). User 1 wrote that working online was both an advantage and disadvantage because on the one hand ‘*you can work on your translation anywhere*’, but on the other hand, it could be a problem for those who have no Internet connection. He added that since Pootle is a web-based system, it certainly contains ‘*a lot of bugs that others softwares do not have*’. Users 2 and 4 affirmed that working online could be an ‘*advantage because you don’t have to install anything and updates of the program are immediate*’, but it is a disadvantage for users who have slow Internet connexion. User 3 answered that ‘*working online is a huge advantage when using software like Pootle since the translation memory is shared among all users*’. He then added that Internet connexion could also be a disadvantage, but the fact that Pootle allows users to export their translation in order to work offline ‘*solves any problem of this kind*’.

In **VQ 29** and **PQ28** we asked participants if there was something they would change in the tools. Two Virtaal evaluators out of four provided their opinion by answering to **VQ29**: User 1 would have changed the tool’s layout and suggestions, while User 4 affirmed that it would have been useful to see translated text directly on the application during its localisation ‘*in order to put things into context*’. For what concerns Pootle (**PQ28**), User 1 affirmed that glossaries should be increased. User 3 wrote that ‘*it would be useful to let
registered users directly create projects on the system’. According to User 4, ‘a percentage rate of matching’ should be added in TM suggestions.

The table below provides a summary of the questionnaires results for Virtaal and Pootle.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Virtaal average score</th>
<th>Pootle average score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>4.36</td>
<td>3.91</td>
</tr>
<tr>
<td>Efficiency</td>
<td>4.75</td>
<td>4.75</td>
</tr>
<tr>
<td>Usability</td>
<td>5.19</td>
<td>5.30</td>
</tr>
<tr>
<td>Average</td>
<td>4.76</td>
<td>4.65</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>0.50</td>
<td>0.70</td>
</tr>
</tbody>
</table>

Table 19: Questionnaires results for Virtaal and Pootle.

Three main characteristics were evaluated: functionality, efficiency and usability. As shown in Table 19, in the evaluation of functionality Virtaal obtained a higher average score (4.36) than Pootle (3.91). It can also be observed that the tools’ efficiency had the same average score of 4.75. Finally, concerning the evaluation of usability, there was a slight difference between the two tools’ average scores: Virtaal had 5.19, while Pootle obtained 5.30. Despite all small differences in questionnaires results, we can affirm that in general there was not a considerable difference between Virtaal and Pootle’s average scores for the three characteristics (Virtaal $\bar{x} = 4.76, sd = 0.50$; Pootle $\bar{x} = 4.65, sd = 0.70$).
7.4. Summary

In this section, we are providing a summary of results for our evaluation of Virtaal and Pootle. The results of the eight experiments are here summarised and compared by the means of the table below.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Sub-characteristics</th>
<th>Means of evaluation</th>
<th>Virtaal average score</th>
<th>Pootle average score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>Suitability</td>
<td>Questionnaire</td>
<td>4.14</td>
<td>3.42</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>Questionnaire</td>
<td>4.58</td>
<td>4.41</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Time behaviour</td>
<td>BB FlashBack, Express</td>
<td>Introduction 5m37s, Translation 22m12s, Revision 0m28s</td>
<td>Introduction 5m38s, Translation 40m46s, Revision 0m54s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Questionnaire</td>
<td>4.75</td>
<td>4.75</td>
</tr>
<tr>
<td>Usability</td>
<td>Understandability</td>
<td>Questionnaire</td>
<td>4.62</td>
<td>4.75</td>
</tr>
<tr>
<td></td>
<td>Learnability</td>
<td>Questionnaire</td>
<td>5.37</td>
<td>5.50</td>
</tr>
<tr>
<td></td>
<td>Operability</td>
<td>BB FlashBack, Express</td>
<td>Total clicks 46.25, Total keystrokes 178</td>
<td>Total clicks 209.75, Total keystrokes 283</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Questionnaire</td>
<td>5.58</td>
<td>5.66</td>
</tr>
</tbody>
</table>

Table 20: Summary of results for Virtaal and Pootle.

By following the indications of the EAGLES final report (sections 4.2.-4.4.), we decided to evaluate three main characteristics in our comparison of Virtaal and Pootle: functionality, efficiency and usability. For each characteristic, one or more sub-characteristics were selected: suitability and accuracy for functionality, time behaviour for efficiency and understandability, learnability, operability and attractiveness for the
evaluation of usability. Table 20 shows results in relation to the evaluated characteristics and sub-characteristics. It can be observed that for both functionality sub-characteristics Virtaal’s average score, obtained through questionnaires, was higher than Pootle’s. Efficiency, and in particular time behaviour, was evaluated analysing both video and questionnaire results. The first showed that participants using Pootle took more time for all three phases of the experiments. It must be said that the only significant gap was detected at the translation stage, and especially during the experiment performed by User P1, but data was here altered by the fact that there was an Internet connection problem and the evaluator had to retranslate some strings. Nonetheless, we have previously demonstrated that if we do not take this user into consideration, Virtaal still gets better results in this variable. On the other hand, Virtaal and Pootle’s average scores were the same for what concerns questionnaires results on time behaviour. For all usability sub-characteristics, apart from operability, Pootle obtained slightly higher average scores than Virtaal. In particular, from the table above it can be observed that questionnaires results for understandability, learnability and attractiveness provided Pootle with higher average scores than Virtaal. On the contrary, results for operability – which were obtained through the examination of BB FlashBack Express videos – showed that the total number of clicks and keystrokes was shorter in experiments with Virtaal, while the number of open windows was the same for both Virtaal and Pootle.

7.5. Findings

In this section we are going to discuss results in relation to our research question: which tool would be better for a novice translator who wants to start collaborating on a FOSS project?

For what concerns results from our objective approach, and in particular time behaviour, we are not taking into consideration the time took for the translation task, since it mainly depended on users’ performance. On the contrary, results on familiarisation time were more interesting for the purpose of our study. In that case, Virtaal and Pootle obtained nearly the same results (Virtaal $\bar{x} = 5m37s$, $sd = 1.86$; Pootle $\bar{x} = 5m38s$, $sd = 1.10$).

Analysing the results obtained for the evaluation of the tools’ operability, it can be noticed that Virtaal was far better than Pootle, since it avoided useless clicks, minimising the use of
the mouse. Although Virtaal may seem the most appropriate tool in this case, both tools have to be taken into consideration instead, since operability largely depends here on the wide availability of shortcuts in Virtaal, which can also be a negative aspect if we consider users who are not used to take advantage of the keyboard.

In the context of results from our subjective approach, we are not taking into consideration functionality, together with its sub-characteristics (suitability and accuracy), since they concerned the tools’ performance rather than participants’ reaction to Virtaal and Pootle. We are focusing on usability sub-characteristics instead. For understandability, Virtaal and Pootle’s results were almost the same, because participants could have used both tools without the provided instructions, maybe by taking more time at the beginning or by reading the documentation.

Concerning learnability, as affirmed by participants to our evaluation, both Virtaal and Pootle are appropriate for beginners, since there is almost no difficulty in learning how to use them.

Results for the evaluation of attractiveness showed that participants enjoyed working with both Virtaal and Pootle. The tools were developed by the same organisation, therefore the only difference in this context consists in choosing to work with a web-based system or a stand-alone tool. But this depends on someone’s task and scope.

7.6. Conclusion

In this chapter, we analysed the data collected in our evaluation. By taking an objective approach (section 7.2.), we focused on videos analysis, from which we could evaluate Virtaal and Pootle’s efficiency through their time behaviour (section 7.2.1.) and operability (section 7.2.2.). By using BB FlashBack Express, we determined the time took for each experiment and task, but also the number of mouse clicks, keystrokes and open windows performed in all tests. The chapter then continues with the analysis of questionnaires results from which we could evaluate the tools’ functionality through suitability and accuracy (sections 7.3.1. and 7.3.2.), their efficiency (section 7.3.3.), and their usability through understandability, learnability and attractiveness (sections 7.3.4.-7.3.6.). In section 7.4., we provided a summary of results obtained from objective and subjective approaches. In conclusion, section 7.5. was centred on the discussion of results in relation to the purpose of our study.
8. Online Survey

8.1. Introduction

This chapter is focused on an online survey (Annex VII) completed by active users of Virtaal and Pootle. First of all we will explain the purpose of this survey (section 8.2.). Then users’ profile will be delineated in order to have an idea of the participants’ background (section 8.3.). In section 8.4., we will describe LimeSurvey, the web-based system used to create and publish our online survey. Afterwards, we will provide the analysis of results by using the same structure as for the previous evaluation, in order to make the comparison more simple (section 8.5.). We will firstly describe results for suitability (section 8.5.1.) and accuracy (section 8.5.2.), then for efficiency (section 8.5.3.) and finally for understandability (section 8.5.4.), learnability (section 8.5.5.) and attractiveness (section 8.5.6.). In section 8.6., we will finally compare the online survey results to those collected from the experiments questionnaires.

8.2. Purpose

Virtaal and Pootle’s active users were at the centre of this online survey. Their opinions were collected in order to make a comparison of the obtained results with those of the previous evaluation questionnaires. The main purpose of this study was therefore to compare different points of view on Virtaal and Pootle, in relation to the level of participants’ experience with the analysed tools. However, for this comparison we are not taking into consideration all results from our previous evaluation, but only those obtained from questionnaires provided to non-experienced users after they performed experiments.

8.3. Users’ Profile

According to Choi et al. (2005: 10), in order to conduct a survey among ‘an educated section of the population (e.g., a professional group) concerning a subject of interest to its members, a mailed questionnaire might be appropriate’. Contributors to our online survey were 25 users of the following mailing lists, obtained from the Pootle servers of different FOSS projects:

- translate-poote@lists.sourceforge.net
- gnome-i18n@gnome.org
Before starting with the real evaluation, contributors to this survey were provided with some general questions aiming at collecting their background information. Participants’ age ranged from 18 to 45 years. A wide variety of native languages was registered within these users. They are listed in the table below, together with the language combinations, in the way they were indicated by participants to the survey.

<table>
<thead>
<tr>
<th>1st Native Language</th>
<th>2nd Native Language</th>
<th>3rd Native Language</th>
<th>Language Combination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afrikaans</td>
<td>English</td>
<td>Afrikaans</td>
<td>English &gt; Afrikaans</td>
</tr>
<tr>
<td>Afrikaans</td>
<td>English</td>
<td>Afrikaans</td>
<td>English &gt; Afrikaans</td>
</tr>
<tr>
<td>Albanian</td>
<td>English</td>
<td>Albanian</td>
<td>English &gt; Albanian</td>
</tr>
<tr>
<td>Basque</td>
<td>Spanish</td>
<td>English</td>
<td>English &gt; Spanish</td>
</tr>
<tr>
<td>Belarusian</td>
<td>Russian</td>
<td>English</td>
<td>English &gt; Russian, Belarusian</td>
</tr>
<tr>
<td>Bulgarian</td>
<td>English</td>
<td>Bulgarian</td>
<td>English &gt; Bulgarian</td>
</tr>
<tr>
<td>Czech</td>
<td></td>
<td></td>
<td>English &gt; Czech</td>
</tr>
<tr>
<td>English</td>
<td>German</td>
<td>Cantonese</td>
<td>English &gt; Scottish Gaelic</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-----------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>English</td>
<td></td>
<td></td>
<td>None⁴¹</td>
</tr>
<tr>
<td>French</td>
<td></td>
<td></td>
<td>French &gt; German, English</td>
</tr>
<tr>
<td>German</td>
<td></td>
<td></td>
<td>English &gt; Scottish Gaelic</td>
</tr>
<tr>
<td>German</td>
<td></td>
<td></td>
<td>German &gt; English</td>
</tr>
<tr>
<td>Hebrew</td>
<td></td>
<td></td>
<td>English &gt; Hebrew</td>
</tr>
<tr>
<td>Hindi</td>
<td>Maithili</td>
<td></td>
<td>English &gt; Hindi, Maithili</td>
</tr>
<tr>
<td>Italian</td>
<td>Croatian</td>
<td></td>
<td>Italian &gt; English</td>
</tr>
<tr>
<td>Italian</td>
<td></td>
<td></td>
<td>English &gt; German, Italian</td>
</tr>
<tr>
<td>Lithuanian</td>
<td>Russian</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marathi</td>
<td>Hindi</td>
<td>English</td>
<td>Marathi &gt; English</td>
</tr>
<tr>
<td>Portuguese</td>
<td></td>
<td></td>
<td>Brazilian Portuguese &gt; Portuguese, Spanish, English, Italian (a little bit)</td>
</tr>
<tr>
<td>Portuguese</td>
<td></td>
<td></td>
<td>English &gt; Portuguese</td>
</tr>
<tr>
<td>Spanish</td>
<td></td>
<td></td>
<td>Spanish, Catalan &gt; English</td>
</tr>
<tr>
<td>Tamil</td>
<td>English</td>
<td></td>
<td>Tamil &gt; English</td>
</tr>
<tr>
<td>Ukrainian</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 21: A list of participants’ Native languages.

As shown in Table 21, there were few participants sharing common native languages: four for English, three for German, and only two for Afrikaans, Hindi, Italian, Spanish, Russian and Portuguese. In addition, we can observe that six contributors out of twenty-five were bilingual, while two other users had even three native languages. When we asked participants to indicate their English level, only 2 of them answered they had a native speaker level. Ten users out of 25 had an advanced English knowledge, nine a proficient, and the remaining four had an intermediate level. An overview concerning participants’ education can be found in the figure below.

---

⁴¹ Later in the survey, we found out that this user is a developer who works on i18n and coordinates l10n for software.
Figure 1 shows that the majority of participants (eleven out of twenty-four) earned a Bachelor’s degree, eight of them had a Master’s degree, while two obtained a High school diploma – which is also the same number of contributors who had no degree at all –, and the remaining user had a Doctoral degree. The participants’ roles in the field of translation were different: fourteen users out of twenty-two added their own answer in the field Other – almost all were volunteers or interested in FOSS localisation as a hobby – five were professional translators and the remaining three were students. All participants had experience in software localisation, in particular 12 users out of twenty-three localised as a hobby and the remaining six worked in this field. Twenty out of twenty-one contributors to this survey affirmed they have been engaged in FOSS projects for longer than 1 year, while the remaining user indicated a period from 6 to 12 months.

8.4. Lime Survey

LimeSurvey is an advanced FOSS system to create powerful and valuable online surveys, thus collecting answers by thousands of users. In general, this program can be installed on the researcher’s web server or locally, but for the implementation of our survey, we decided to create an account on LimeService.com, an online hosting platform allowing users to easily generate and run intuitive online surveys (LimeSurvey Manual, 2014). This service is
available at no cost, but each subscriber can only collect up to 25 free answers per month. Once our registration was completed, we received the following link to log in to our installation: http://etuunige.limequery.org/admin.

Fig. 12: LimeSurvey administrator installation interface on LimeService.com.

The figure above shows the installation interface for our online survey, which is a sort of administrator dashboard to quickly create, manage and publish surveys. The general settings for our survey included the following specifications:

- English was set as the base language for the survey, because it is usually used to communicate in the FOSS community.
- Questions were presented group by group. They were almost identical to those of Virtaal and Pootle’s final questionnaires for our previous evaluation. There were 43 questions in total: 16 general questions on background information, 14 specific questions about Pootle and 13 specific questions about Virtaal.
- Questions numbers and a progress bar were provided in order to let contributors know their progression with the survey.
- Timings were stored in a separate table automatically generated by LimeSurvey. In this way, we could know the time took by users to complete each question, as well as the whole survey.
- The ‘save and resume later’ option was activated. Since our survey guaranteed anonymity, we added this option so that participants could save their given answers
and continue the survey afterwards. In this way, they could quit the survey page without the risk of losing their work.

8.5. Online Survey Analysis

Our online survey started on the 3rd November 2014 at midnight and expired on the 30th November 2014 at 11:59 pm. Contributors were provided with the following URL giving access to the survey: http://etuunige.limequery.org/index.php/296662/lang-en.

The figure below shows the survey home page where the subject and purpose of our study were explained in order to provide users with some information about what they were exactly going to do.

![Online survey introductory webpage.](image)

For what concerns time, we followed the suggestion provided by Choi et al. (2005: 7), stating that ‘self-administered questionnaires typically take 10 to 20 minutes to complete’. We therefore executed some tests in order to fix time so that contributors’ involvement did not exceed 20 minutes. As shown in the figure below, users participating to our survey were twenty-five in total, but only eighteen of them answered to all questions, the other seven left some parts incomplete.
We are now going to analyse the results obtained from this online survey, representing Virtaal and Pootle’s active users’ subjective evaluation. As performed in our previous study, we are here evaluating three characteristics, as well as their sub-characteristics. They were the same evaluated in our experiments: functionality, including suitability and accuracy, efficiency and usability, consisting of understandability, learnability and attractiveness.

8.5.1. Suitability

Virtaal and Pootle’s suitability was evaluated through contributors’ opinions on the interface and features of the tools. Two questions were asked for Pootle (OSQ19 and OSQ20) and two for Virtaal (OSQ33 and OSQ34).

In OSQ19 and OSQ33, we asked users to give a score from 1 (very poor) to 6 (excellent) to a list of features of Pootle and Virtaal GUI. As shown in the table below, Pootle obtained an higher score for the following features: layout (Pootle $\bar{x} = 4.45$, $sd = 0.75$; Virtaal $\bar{x} = 4.43$, $sd = 1.09$), intuitiveness (Pootle $\bar{x} = 4.50$, $sd = 0.94$; Virtaal $\bar{x} = 4.43$, $sd = 1.31$), familiarity (Pootle $\bar{x} = 4.80$, $sd = 0.89$; Virtaal $\bar{x} = 4.60$, $sd = 1.18$) and design (Pootle $\bar{x} = 4.20$, $sd = 1.05$; Virtaal $\bar{x} = 4$, $sd = 1.09$). On the other hand, Virtaal had a better result than Pootle for the other two interface features: clarity (Virtaal $\bar{x} = 4.56$, $sd = 1.09$; Pootle $\bar{x} = 4.55$, $sd = 0.94$) and responsiveness (Virtaal $\bar{x} = 4.68$, $sd = 0.87$; Pootle $\bar{x} = 4.20$, $sd = 1.19$).
<table>
<thead>
<tr>
<th>FEATURE</th>
<th>VIRTAAL</th>
<th>POOTLE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Layout</strong></td>
<td>$\bar{x} = 4.43$</td>
<td>$\bar{x} = 4.45$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.09$</td>
<td>$sd = 0.75$</td>
</tr>
<tr>
<td><strong>Clarity</strong></td>
<td>$\bar{x} = 4.56$</td>
<td>$\bar{x} = 4.55$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.09$</td>
<td>$sd = 0.94$</td>
</tr>
<tr>
<td><strong>Intuitiveness</strong></td>
<td>$\bar{x} = 4.43$</td>
<td>$\bar{x} = 4.50$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.31$</td>
<td>$sd = 0.94$</td>
</tr>
<tr>
<td><strong>Familiarity</strong></td>
<td>$\bar{x} = 4.60$</td>
<td>$\bar{x} = 4.80$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.18$</td>
<td>$sd = 0.89$</td>
</tr>
<tr>
<td><strong>Responsiveness</strong></td>
<td>$\bar{x} = 4.68$</td>
<td>$\bar{x} = 4.20$</td>
</tr>
<tr>
<td></td>
<td>$sd = 0.87$</td>
<td>$sd = 1.19$</td>
</tr>
<tr>
<td><strong>Design</strong></td>
<td>$\bar{x} = 4$</td>
<td>$\bar{x} = 4.20$</td>
</tr>
<tr>
<td></td>
<td>$sd = 1.09$</td>
<td>$sd = 1.05$</td>
</tr>
</tbody>
</table>

Table 22: Summary of participants’ answers to OSQ19 and OSQ33.

Also for **OSQ20** and **OSQ34**, participants had to give a score on a scale of 1 to 6, but this time to Virtaal’s features in relation to their usefulness in the localisation process. As shown in the table below, Virtaal obtained considerably higher scores than Pootle for all the following features: navigation (Virtaal $\bar{x} = 4.18$, $sd = 1.10$; Pootle $\bar{x} = 4.05$, $sd = 1.14$), translation memory (Virtaal $\bar{x} = 4.31$, $sd = 1.44$; Pootle $\bar{x} = 3.55$, $sd = 1.14$), terminology help (Virtaal $\bar{x} = 4$, $sd = 1.36$; Pootle $\bar{x} = 3.70$, $sd = 1.26$), quality check (Virtaal $\bar{x} = 4.37$, $sd = 1.45$; Pootle $\bar{x} = 3.85$, $sd = 1.13$), web look-ups (Virtaal $\bar{x} = 3.75$, $sd = 1.29$; Pootle $\bar{x} = 3.15$, $sd = 1.08$) and placeable manipulation (Virtaal $\bar{x} = 4.25$, $sd = 1.52$; Pootle $\bar{x} = 3.70$, $sd = 1.41$).
8.5.2. Accuracy

Accuracy was evaluated by the means of two questions for both Pootle (OSQ17 and OSQ18) and Virtaal (OSQ31 and OSQ32), collecting participants’ opinions on the tools’ performance in relation to the localisation process.

Contributors answered to OSQ17 and OSQ31 indicating to which extent they agreed with the fact that the tool in question did whatever they needed it to do in the localisation process. Pootle had an average score of 4.40 \((sd = 1.14)\). However, four aspects were highlighted by contributors:

1. Working with Virtaal is preferable when dealing with projects having ‘less than a dozen or so source files’.
2. ‘Dealing with some formats from the .NET world is poorly supported at this stage.’
3. Pootle’s users cannot ‘customize the output format of a string, which sometimes is necessary to adjust date & time in some locales’.
4. ‘In the proofreading process, Pootle doesn’t provide a mechanism to detect misleading words, double spaces, bad punctuation and so on.’

For what concerns Virtaal, it obtained an average score of 4.87 \((sd = 1.02)\). A user wrote a comment underlying that complementary tools should be used in order ‘to ensure quality checks, like the translate-toolkit tool’.

In OSQ18, eleven users out of 20 agreed with the statement saying that the tool reliability was satisfying, seven slightly agreed and the remaining two disagreed \((Pootle \bar{x} = 4.35, sd = 0.93)\). A contributor indicated that Pootle was concerned with ‘slight issues of occasional data’.
loss and small bugs’. By answering to OSQ32, nine participants out of 16 agreed, four slightly agreed, two completely agreed and the last one slightly disagreed (Virtaal $\bar{x} = 4.75$, $sd = 0.77$). Contributors’ comments on Virtaal reliability suggested that the ‘handling of placeholders by the TM could be improved’ and indicated that the tool was not available on OS X and it was ‘not very reliable on Windows’.

8.5.3. Efficiency

Efficiency was evaluated analysing results of two questions for Pootle (OSQ21 and OSQ22) and two for Virtaal (OSQ35 and OSQ36).

In OSQ21 and OSQ35, we asked if the tool took too much time to get something done. It must be specified that in this case we had to adapt the scale by following the other scores, where 6 represented the best and 1 the worst. Therefore, here completely agree referred to 1 and completely disagree to 6. Pootle obtained an average score of 3.94 ($sd = 1.17$), which was considerably lower than Virtaal’s 4.37 ($sd = 1.54$). By answering to OSQ21, two users wrote further comments to complete their answers: one specified that all depended on the server implementation, while the other affirmed that Pootle was ‘very practical and accelerate[d] the translation process’. For what concerns Virtaal, a user explained that he slightly agreed because ‘Virtaal seems confusing to newcomers in localisation projects and to some veterans too, due it’s spartan user interface’.

In OSQ22 and OSQ36 we asked evaluators to indicate to what extent they agreed on the fact that the tool improved the localisation process. Virtaal obtained a slightly higher score ($\bar{x} = 5.25$, $sd = 0.68$) than Pootle ($\bar{x} = 5.10$, $sd = 0.80$). Three comments for answers to OSQ22 were provided. A user affirmed that Pootle certainly improve[d] the localisation process ‘over Trados which is overengineered and over-hyped’, another one specified that this was especially true ‘for people with little technical skill’ and the last one underlined that ‘[t]ools just provide a helping hand but the people involved are the point to consider’.

8.5.4. Understandability

A question was used to evaluate Virtaal and Pootle’s understandability: OSQ25 for Pootle and OSQ39 for Virtaal. Participants were asked to indicate if the tools’ documentation and online help are useful. Pootle and Virtaal’s scores were nearly the same (Pootle $\bar{x} = 4.29$, $sd = 0.77$; Virtaal $\bar{x} = 4.28$, $sd = 0.72$). By answering to OSQ25, a contributor wrote that ‘the
separation of User, Sysadmin, Translation admin roles needs to be a bit cleaner'. On the other hand, for OSQ39 five participants specified they had never used documentation and another one affirmed he did not need it, because he relied on his ‘previous knowledge and intuition on using localisation software’.

8.5.5. Learnability

In order to evaluate the tools’ learnability, we asked two questions for both Virtaal (OSQ37 and OSQ38) and Pootle (OSQ23 and OSQ24).

In OSQ37 and OSQ23, participants were asked to evaluate the difficulty encountered in learning how to use the tool on the basis of their experience, on a scale of 1 to 6, where the latter represented the highest level of difficulty. Here again we adapted the scale to other scores, by attributing 6 to the lowest level of difficulty (1) and 1 to the highest (6). Pootle obtained a higher score ($\bar{x} = 4.84, sd = 1.16$) than Virtaal ($\bar{x} = 4.50, sd = 0.96$).

In OSQ38 and OSQ24, we asked participants if, in their opinion, the tools are appropriate for beginners in software localisation. For what concerns Pootle 18 users out of 19 answered yes, the last one no. Two comments were provided in addition to these answers: a contributor specified that a ‘quick-start guide’ is although needed, another one wrote that ‘the ideal combination would be using Pootle and also using command line translation/localisation tools to strengthen the translation’.

13 users out of 16 replied positively to OSQ24, the remaining three answered no.

On one hand was underlined that even if Virtaal ‘contains a lot of advanced functionality, a wide selection of features are very easy to use’. On the other hand, a user affirmed that poEdit would be a better alternative than Virtaal for beginners, although this latter ‘should not be discarded’.

8.5.6. Attractiveness

In order to evaluate attractiveness, indicating users’ general satisfaction with the tools, five questions were formulated for Pootle (OSQ26, OSQ27, OSQ28, OSQ29, OSQ30) and four for Virtaal (OSQ40, OSQ41, OSQ42, OSQ43).

By answering to OSQ26 and OSQ40, almost all participants affirmed they enjoyed working with the tools. For what concerns Pootle, 16 users out of 19 answered yes, one replied no but he did not give a reason for his answer, while the remaining two did not answer to the question. One of the users also underlined that although with Pootle ‘you can
only translate what is on someone’s server’, ‘it works well for what it is built for’. In OSQ40, 15 users out of 18 affirmed they enjoyed working with Virtaal. Then, as for Pootle, two participants did not answer to the question and one replied negatively by adding that he preferred Pootle because no installation was required.

In OSQ27, participants were asked to indicate the best thing about working with Pootle. Answers to this question were various and we summarised them by the following points:

- It is a FOSS tool.
- It is fast.
- There is no need for installation on local machine.
- Users do not need to ‘learn too many technical skills’, since it ‘provides a comprehensive user interface, which is friendly and makes the translation of a large application more funny to do’.
- It allow to ‘navigate over a complex project’ containing multiple L10n files.
- It provides users with translation statistics. One of the participants to the survey underlined that the latter are more than a simple progress bar, they are ‘an indicator of motivation’.

By answering to OSQ41, participants indicated six best things about working with Virtaal: speed, minimal interface, shortcuts, networked/local TM and terminology, ‘fuzzy translation memory matched that show changes to the strings’ and ‘real-time quality check’.

In OSQ28 and OSWQ42, we asked for the most annoying thing about working with the tools. Concerning Pootle, participants indicated several negative aspects. From a general point of view, one of the contributors affirmed that Pootle was not ‘as interactive as a local application’, while another one underlined that due to the simplicity of the tool, ‘the translator can feel more relaxed than normally (s)he would be and carelessly made mistakes with bad punctuation, misleading words, repeated words, and so on’. From a technical point of view, the annoying things designated by participants concerned the lack of csv support and of ‘advanced mechanisms to handle with special treatment of variables formatters’. In addition, a user affirmed that TM should be improved and he also suggested adding a feature to combine ‘TM results from local and remote locations’.

For what concerns Virtaal, annoying things indicated by contributors were summarised in the following list:
- It is not ‘available in all platforms reliably’.
- User information is not stored.
- It does not ‘break up multi-sentence chunks into individual sentences in-program and then reassemble for saving the file’.
- Lack of support for projects with multiple L10n files.
- Lack of ‘search filter for source or translation only’.
- It has a ‘weird layout’, since ‘the fields appear to be so much grouped’ and there is not a real separator line between the source and target strings.

OSQ29 and OSQ43 asked if Virtaal and Pootle are good software localisation tools. 17 participants out of 19 answered positively to OSQ29, one replied no without giving a reason for his answer and the remaining one did not answer. For what concerns Virtaal, 16 users out of 18 answered positively and the remaining two did not reply to OSQ43. In addition, two comments were provided: on one hand, a user affirmed that for FOSS localisation Virtaal was ‘maybe the best’, on the other hand another one argued that ‘it lacks certain important functionalities that Pootle has’.

An extra question, which was specific to Pootle, asked participants to indicate if working online is an advantage or disadvantage (OSQ30). According to participants’ answers, working online is an advantage for essentially two reasons: on one hand it allows users to work from different computers and to access files beyond physical limits, on the other hand it ‘helps collaborating’, as a matter of fact it facilitates team work and communication.

For what concerns the negative aspects of working online, two main disadvantages were highlighted: first of all, users are not able to open files from their own systems, and then ‘for some locales, connectivity is a real issue’ – for example if some users ‘live in a country where access to Internet is not always available, is expensive or unreliable. One of the participants suggested a solution to this latter disadvantage concerning online work: Pootle’s users should be able to work ‘even without an active Internet connection’; in this way, their translations could be ‘synchronized back to the server’ once connectivity is available again. But finally the participant acknowledged that ‘this would potentially create conflicts if other people have been actively working on projects at the same time’.

At the end of this online survey, after the specific questions, we had provided participants
with an empty field were they could write any other feedback or remark. Only one of them used it in order to give his general opinion on the subject of the survey, which is entirely quoted below:

‘My opinion as coordinator of a FOSS translating team is that the great translating environment is which uses as many tools as it is available and which has a collaborating and active members. Pootle could be used to accelerate the translation process and simultaneously among different translators. The translation file generated at the end of a Pootle workflow could be reviewed using Virtaal by a proofreader and as a final step and quality assurance, some command line tools like msgfmt and translate-toolkit (polfilter) could be used to ensure the translation file does not have broken tags, bad punctuation, misspelled words, and so on.’

The table below provides a summary of the online survey results for Virtaal and Pootle.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Virtaal average score</th>
<th>Pootle average score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>4.55</td>
<td>4.20</td>
</tr>
<tr>
<td>Efficiency</td>
<td>4.81</td>
<td>4.52</td>
</tr>
<tr>
<td>Usability</td>
<td>4.96</td>
<td>5.09</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>4.77</strong></td>
<td><strong>4.60</strong></td>
</tr>
<tr>
<td><strong>Standard deviation</strong></td>
<td><strong>0.20</strong></td>
<td><strong>0.45</strong></td>
</tr>
</tbody>
</table>

Table 24: Summary of online survey results for Virtaal and Pootle.

Also for this evaluation, three main characteristics were taken into consideration: functionality, efficiency and usability. Table 24 shows that in the evaluation of functionality and efficiency, Virtaal obtained higher average scores (respectively 4.55 and 4.81) than Pootle (4.20 and 4.52). Finally, for what concerns usability, there was a slight difference between the two tools’ average scores: Virtaal had 4.96, while Pootle obtained 5.09. Overall, despite these differences between Virtaal and Pootle, it must be underlined that there was not a substantial difference between the tools’ average scores for all three characteristics (Virtaal $\bar{x} = 4.77$, $sd = 0.20$; Pootle $\bar{x} = 4.60$, $sd = 0.45$).
8.6. Results comparison

In this section, we are analysing results from novice translators’ questionnaires with those from active users’ answers to our online survey on Virtaal and Pootle. A comparison between them is shown in the table below.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Sub-characteristics</th>
<th>Virtual average score</th>
<th>Pootle average score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Novice translators</td>
<td>Active users</td>
</tr>
<tr>
<td>Functionality</td>
<td>Suitability</td>
<td>4.14</td>
<td>4.29</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>4.58</td>
<td>4.81</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Time behaviour</td>
<td>4.75</td>
<td>4.81</td>
</tr>
<tr>
<td>Usability</td>
<td>Understandability</td>
<td>4.62</td>
<td>4.28</td>
</tr>
<tr>
<td></td>
<td>Learnability</td>
<td>5.37</td>
<td>4.78</td>
</tr>
<tr>
<td></td>
<td>Attractiveness</td>
<td>5.58</td>
<td>5.84</td>
</tr>
</tbody>
</table>

Table 25: Comparison between evaluation and online survey results.

Here again three main characteristics, together with their sub-characteristics, are evaluated: functionality (suitability and accuracy), efficiency (time behaviour) and usability (understandability, learnability, and attractiveness). Table 25 shows results in relation to these characteristics and sub-characteristics and to the evaluators, i.e. novice translators who completed the final questionnaires in our experiments and active users of Virtaal and Pootle who performed the online survey. It can be observed that Virtaal obtained higher average scores than Pootle for suitability and accuracy in both novice translators’ and active users’ answers.

For what concerns time behaviour, Virtaal and Pootle obtained the same average score in questionnaires, while online survey results showed that Virtaal had an higher average score ($\bar{x} = 4.81$, $sd = 0.62$) than Pootle ($\bar{x} = 4.52$, $sd = 0.82$).

For all three usability sub-characteristics (understandability, learnability and attractiveness), in questionnaires completed by novice translators Pootle obtained higher average scores than Virtaal. In online survey results, it can be observed that Virtaal and Pootle had nearly
the same average score for understandability (Virtaal $\bar{x} = 4.28$, $sd = 0.72$; Pootle $\bar{x} = 4.29$, $sd = 0.77$). For learnability active users rated Pootle better ($\bar{x} = 5.28$, $sd = 0.62$) than Virtaal ($\bar{x} = 4.78$, $sd = 0.39$), while for attractiveness Virtaal obtained a slightly higher score ($\bar{x} = 5.84$, $sd = 0.22$) than Pootle ($\bar{x} = 5.71$, $sd = 0.01$).

8.7. Conclusion

This chapter was centred on an online survey completed by active users of Virtaal and Pootle. We have firstly clarified its purpose (section 8.2.) and then information on participants’ background was provided, by analysing their answers to some general questions (section 8.3.). In section 8.4., we have described LimeSurvey and have specified its settings for our survey. Afterwards, we have delineated the survey implementation and we have provided results by using the same structure as for our previous evaluation (section 8.5.). Functionality, efficiency and usability were evaluated through the results obtained for their sub-characteristics, respectively suitability (section 8.5.1.) and accuracy (section 8.5.2.), time behaviour (section 8.5.3.), understandability (section 8.5.4.), learnability (section 8.5.5.) and attractiveness (section 8.5.6.). In section 8.6., we compared non-experienced users’ questionnaires results with those from our online survey performed by active users of Virtaal and Pootle.
9. Conclusions and Future Work

This thesis was focused on FOSS localisation, and, in particular, on the analysis and comparison of the two most used programs in FOSS localization projects: Pootle, a web-based system, and Virtaal, a stand-alone tool. The main purpose of our study was to figure out which system would be better for a non-professional translator who wants to start collaborating on a FOSS project.

By following the EAGLES final report, we decided to evaluate three main characteristics: functionality, efficiency and usability. These latter were broken down into one or more sub-characteristics in order to become measurable and comparable. Suitability and accuracy were examined for the evaluation of functionality, time behaviour for efficiency and understandability, learnability, operability and attractiveness for usability.

Time behaviour was evaluated through the analysis of both video and questionnaire. Results from video analysis showed that participants using Pootle took more time for all the phases of the experiments, even if we did not take into consideration the user who had an Internet connection problem. On the other hand, questionnaires results showed that Virtaal and Pootle had exactly the same average score for efficiency. It must be said that the main difference observed from Virtaal and Pootle’s video analysis concerned the stage of the translation task, but this result could have been depended on the participants’ translation skills. Since the purpose of our study was not to evaluate participants’ performance, we should focus on the familiarisation time. For this latter, participants’ took nearly the same time with both Virtaal and Pootle. Therefore we can affirm that both tools obtained good results in relation to the scope of this study.

Results for operability – which were obtained through the analysis of videos – showed that the total number of clicks and keystrokes was shorter in experiments with Virtaal. This was probably because Virtaal normally provides users with more shortcuts than Pootle, as it was designed to minimise the use of the mouse, which is a very positive aspect, since, as we could observe from the videos, participants’ efficiency was undoubtedly increased and their localisation task was somehow simplified. For what concerns the number of open windows, it was identical for Virtaal and Pootle.

Both suitability and accuracy were evaluated by the means of questionnaires. Results indicated that Virtaal obtained a higher score than Pootle, probably because online systems are in general less stable than desktop tools.
Finally, for what concerns understandability, learnability and attractiveness, Pootle scored better than Virtaal according to questionnaires results.

In response to the central question of this thesis, it can be affirmed that both Virtaal and Pootle are appropriate for novice translators who would like to start contributing to FOSS projects. Results, in fact, showed us that if on one hand Virtaal has some important features that Pootle lacks (i.e., auto-completion, auto-correction, numerous shortcuts) and vice versa (i.e., alternative source language, comment option), on the other hand we should not forget the research question of our study and consequently its main purpose, which consisted in choosing the most appropriate tool for beginners in FOSS localisation. In this sense, we can conclude that there were not significant differences between Virtaal and Pootle’s scores and beginners could easily use both. In addition, it can be affirmed that the two tools are even complementary in some ways, since users could work with Pootle for online localisation and with Virtaal when they have no access to Internet.

A separate chapter was written about an online survey completed by active users of Virtaal and Pootle. We decided to conduct this survey, because we thought it would be very interesting to compare non-experienced evaluators’ views with those of active contributors to FOSS projects who constantly use the two tools. Despite all the differences pointed out in our study, we can affirm that in general Pootle and Virtaal scores were quite similar in both our evaluation and online survey. This is certainly due to the fact that the two tools are very similar in every respect, since they were created by the same developers.

Three limitations can be found in our study. First of all, the major limitation is that the number of participants was exiguous, therefore the collected data had no external validity. In addition, the users’ variance might have highly influenced results. The solution could have been implementing the experiments so that the eight participants used both tools on an equivalent task. Half of them could have started with Virtaal and half with Pootle in order to avoid the learning effect. Another limitation is time, since this study was conducted in the context of a Master thesis in Translation Studies. FOSS tools systematic releases and frequent uploads are also a limitation, especially for Pootle, which has been recently released in its version 2.7.0. For future work, experiments should be repeated in order to find out if some features or functionalities were improved.

It would also be interesting to repeat the experiments with a greater number of participants or with different language combinations in order to find out if results would
In addition, for what concerns the online survey, a larger study should be conducted by involving numerous users. Perhaps asking them to collaborate in the evaluation design and implementation could be a good idea to stimulate their interest, since, as said before, collaborative work is at the basis of FOSS philosophy.
Bibliography


Schäler, R. (2013), *Enabling the Global Conversation in Communities*, Localisation Research Centre (LRC), University of Limerick.


Annex I – Information Sheet, Consent Form

Participants Information Sheet
A Comparative Study of two FOSS Localisation Tools:
Pootle and Virtaal

GENERAL INFORMATION
The study in which you are being asked to participate is focused on non-experienced translators’ reaction to two Free and Open Source Software Localisation Tools: Pootle, a web-based system, and Virtaal, a stand-alone tool. You will be asked to localise a small software application with one of the tools and to complete a questionnaire.

PURPOSE OF THE STUDY
The purpose of this study is to figure out which of the concerned tools would be better for a non-professional who wants to start contributing to FOSS projects.

PARTICIPATION
Your participation is voluntary and you are free to withdraw from the study at any time.

DURATION
Participants’ involvement will not exceed 1 hour.

BENEFITS AND RISKS
We cannot guarantee that participants will have some benefits from this study, however possible benefits may include: the acquisition of basic knowledge in the field of Software Localisation, an introduction to the FOSS Localisation world, and learning how to use a software localisation tool while testing it. There are no risks associated with participating in the study.

CONFIDENTIALITY
All information you provide us will be kept confidential. Under no circumstances will identifiable responses be provided to any other third party. No reference will be made in the researcher’s thesis which could link participants to the study.

FURTHER INFORMATION

If you have any questions or concerns, please contact Valeria Siano at: valeria.siano@etu.unige.ch.
CONSENT FORM

Title of the study: A Comparative Study of two FOSS Localisation Tools: Pootle and Virtaal

Researcher: Valeria Siano, FTI, Master of Arts in Translation, Concentration in Translation Technologies

Supervisors: Lucía Morado Vázquez, Marianne Starlander

Please tick the following boxes if you are ready to start.

☐ I confirm that I have read and understood the information sheet for the above study. I have had the opportunity to consider the information, ask questions to the researchers who have answered satisfactorily.

☐ I understand that my participation is voluntary and that I am free to withdraw from the study at any time, without having to give a reason and without any consequence.

☐ I understand that data collected throughout this study may be stored and processed by the researchers.

☐ I understand that any information recorded in the study will remain confidential and no information that identifies me will be made publicly available.

Participant’s name __________________________ Date __________________________ Signature __________________________

Researcher’s name __________________________ Date __________________________ Signature __________________________
Annex II – General Questionnaire

GENERAL QUESTIONNAIRE

Please answer the following questions by filling in the blanks or ticking the appropriate box.

First Name: _________________________ Last Name: _________________________

1. Age:

________________________

2. Native Language(s):

________________________  ________________________  ______________________

EDUCATION

3. What is the highest degree you have earned?

☐ High School diploma
☐ Bachelor’s degree
☐ Master’s degree
☐ Doctoral degree
☐ No diploma or degree
☐ Other (please specify): _________________________

4. English level:

☐ Native speaker
☐ Proficient
☐ Advanced
☐ Intermediate
Elementary
□ No knowledge

COMPUTER SKILLS

5. How often do you use a computer or a similar device?

□ At least once a day
□ 5-6 times a week
□ 2-4 times a week
□ At least once a month
□ Less than once a month
□ Almost never

6. In which context do you use it? (more than one answer possible)

□ Education
□ Work
□ Research
□ Communication
□ Free time
□ Other (please specify): ____________________________

7. Which Operating System do you know? (more than one answer possible)

□ Windows
□ Macintosh
□ Linux
□ Other (please specify): ____________________________

8. Which one do you normally use? (more than one answer possible)

□ Windows
□ Macintosh
□ Linux
□ Other (please specify): ____________________________

9. Which Software Application do you know? (more than one answer possible)

Word Processors:

□ Microsoft Word
□ Apple Pages

129
10. Which one do you regularly use? (more than one answer possible)

Word Processors:
- Microsoft Word
- Apple Pages
- OpenOffice Writer
- Google Docs
- Other (please specify):

Spreadsheets:
- Microsoft Excel
- Apple Numbers
- OpenOffice Calc
- Gnumeric
- Other (please specify):

Web Browsers:
- Internet Explorer
- Safari
- Firefox
- Google Chrome
- Other (please specify):

E-mail Clients:
- Microsoft Outlook
- Eudora
- Mozilla Thunderbird
- Gmail
- Other (please specify):
☐ Safari  
☐ Firefox  
☐ Google Chrome  
☐ Other (please specify): ____________________________

E-mail Clients:

☐ Microsoft Outlook  
☐ Eudora  
☐ Mozilla Thunderbird  
☐ Gmail  
☐ Other (please specify): ____________________________

11. Have you ever taken one or more Translation Technology courses?

☐ Yes  
☐ No

12. If yes, which?

______________________________

13. Do you feel comfortable about your ability to work with computer technologies?

☐ Yes  
☐ No

**USER SECTION**

14. What is your role in the field of translation?

☐ Professional translator  
☐ Student  
☐ Other (please specify): ____________________________

15. What is your language combination?

____________________________________

16. Have you experience in Software Localisation?

☐ Yes, I work (also) in this field  
☐ Yes, I made an/some internship(s) in this field
□ Yes, I took a/some Software Localisation course(s)
□ Yes, I localise software as hobby
□ I have no experience in this field
□ Other (please specify): __________________________

17. Have you ever heard about Free and Open Source projects before?
   □ Yes
   □ No

18. Do you know any Free and Open Source Software?
   □ Yes
   □ No

19. If yes, which?
   _____________________________________________

20. Do you know any CAT tools?
   □ Yes
   □ No (end of the questionnaire)

21. If yes, which?
   _____________________________________________

22. Do you know how to use it/them?
   □ Very well
   □ A little bit
   □ Not very well
   □ Not at all

23. How often do you use it/them?
   □ At least once a day
   □ 5-6 times a week
   □ 2-4 times a week
   □ At least once a month
   □ Less than once a month
   □ Almost never
Annex III – Virtaal Instructions

INSTRUCTIONS
- Virtaal -

Please follow the instructions included in this sheet. You can also find them as Instructions.pdf in your folder on the desktop, so that you can read on screen or click on links. If you get lost or you have any question, please ask.

The entire experiment will be recorded with BB FlashBack, a screen recorder that will let us extract and analyse some data. Participants’ personal data will not be disclosed.

You will translate the GUI (Graphic User Interface) of NoteFly, a small program which is free and open source. Have a look at the running application while translating: open NoteFly from your desktop. You will then need to right click on the icon in the notification area, on the right side of the taskbar.

As for most FOSS projects, you will work with the Gettext system, that is to say that translatable text has been extracted from the source code and it can be found in a text file, known as POT file (Portable Object Template). From this latter, a PO file (i.e., a bilingual text file) has been generated for each target language.

To work with the concerned PO file you will need to use Virtaal, a free and open source CAT tool developed by Translate.org.za. You will find the program on your desktop. Open it and have a look at the dashboard. Here you can find helpful links, providing information about Virtaal, or have access to your recent files.
You will now need to introduce your personal data (that will then appear in the header of the PO file).

Go to **Edit>Preferences**, write the name indicated in your folder on the desktop and your language team (**Italian Team**), as shown in the figure below. You can leave the E-mail address field blank.

![Virtaal Preferences](image)

**Before you start.**

Virtaal is designed to encourage you to work as much as possible with your keyboard, in order to increase your speed. Here is a list of some useful tips that will save you some time:

- After typing your translation, you just need to press **Enter** to go to the next string, while if you have to start a new line, you can press **Shift+Enter**.

- When you need to copy the text from source to target, press **Alt+down**.

- Virtaal will try to complete some long words with **Auto-completion**. Press **Tab** to accept, or simply continue typing if the suggested completion is not what you want. If you wrongly accept something, you can press **Ctrl+Z** to undo.
Virtaal allows you to connect projects with a translation memory or terminology database. It can normally be done by clicking on Edit>Plug-ins, but in this case, we have already activated plug-ins, so you will not need to do it. You will notice that suggestions from translation memory will be displayed under the editing area as shown in the figure below.

(Translate, 2013)

You can accept suggestions provided during your work with a double click, or by pressing Ctrl+1. If there is more than one suggestion, you can press Ctrl+2, Ctrl+3 and so on, respectively for the second, the third, etc..

If you would like to look for a word or an expression on the Web, you can do it directly from Virtaal through web-lookups and avoid copying and pasting. As shown in the figure below, you just need to select words or expressions from the source or target text, right click, and then go to Search to choose a web-lookup.
You are now ready to start localising with Virtaal. Remember to pay attention to all the special elements seen during the short Introduction to Software Localisation: **numbers, acronyms, e-mails, URLs, variables, tags, hot keys and shortcuts.** In Virtaal, they are called **placeables** and the tool puts them in **red**, so that they can be easily recognised and inserted into the target string.

**Localising NoteFly.**

Go to **File > Open > Your Folder** and choose the file with your language code. In your case, it will be **it.po**.
After opening the file, translatable strings will be shown, with your cursor in the field below the source text. First of all, make sure that the language combination in the right bottom corner is correct (English » Italian). Since some strings have already been translated, you will also need to activate the Incomplete mode, so that you can quickly move between untranslated and fuzzy strings to work on them. Go to Navigation, under the menu bar on the left top of the window, and select Incomplete. At any time, you will be able to check the status of the string you are translating, which is displayed on the right, next to the target string.

You can now start localising NoteFly.

! Please note that the 6th string, which is marked as Needs work, has to be revised.

The Quality Check box on the right top of the string displays some remarks saying that something is missing in the target string. Click on it to have more details.
As you might notice, the placeable \{\theta\} is missing. With your cursor in the target string, press Alt+Right to select it, and Alt+Down to insert it directly in the translation. Do the same with all the other placeables you will find while localising.

When all the strings are translated, Save your work and Quit Virtaal.

You have finished the experiment. Please, complete the Final Questionnaire.
Annex IV – Pootle Instructions

INSTRUCTIONS
- Pootle -

Please follow the instructions included in this sheet. You can also find them as Instructions.pdf in your folder on the desktop, so that you can read on screen or click on links. If you get lost or you have any question, please ask.

The entire experiment will be recorded with BB FlashBack, a screen recorder that will let us extract and analyse some data. Participants’ personal data will not be disclosed.

You will translate the GUI (Graphic User Interface) of NoteFly, a small program which is free and open source. Have a look at the running application while translating: open NoteFly from your desktop. You will then need to right click on the icon in the notification area, on the right side of the taskbar.

As for most FOSS projects, you will work with the Gettext system, that is to say that translatable text has been extracted from the source code and it can be found in a text file, known as POT file (Portable Object Template). On the basis of this latter, a PO file (i.e., a bilingual text file) has been generated for each target language.

To work with the concerned PO file you will need to use Pootle, a free and open source web-based system for localisation and translation management developed by Translate.org.za. You can find helpful information about this program at http://pootle.readthedocs.org/en/latest/users/getting_started.html.
Copy http://pootle.sympa.org/ in the address bar of your web browser or simply click on the link to access Pootle. Then click on Log in on the right side of the top menu bar and fill in your credentials. We have created a Pootle account for you, so you will find your user name and password in a .txt file in your folder on the desktop. Select English as the Interface Language and log in.

First of all you need to change your language preferences. Click on the icon shown in the figure below to access your dashboard.

![Dashboard](http://pootle.sympo.org/figure.png)

Go to Settings and select French in the field Alternative source languages. Save the changes and go back to the main page clicking on the Translate Sympa icon on the left top corner.

**Before you start.**

Here is a list of some useful tips that will save you some time:

- After typing your translation, you just need to press Ctrl+Enter to save your changes and go to the next string, while if you have to start a new line, you can press Shift+Enter.

- When you need to copy the text from source to target, click on .
If you would like to look for a word or an expression on the Web, you can do it directly from Pootle through **web-lookups** and avoid copying and pasting. You just need to select words or expressions from the source text and then click on a web-lookup icon – for example, ![Wikipedia](https://en.wikipedia.org) to access Wikipedia.

Pootle connects projects with a **translation memory** and a **terminology database** in order to preserve the coherence of your translation. You will notice that suggestions from translation memory will be displayed under the editing area, while terminology suggestions will be in a box on the left. You can accept suggestions provided during your work with a click as shown in the figure below.

You are now ready to start localising with Pootle. Remember to pay attention to all the special elements seen during the short Introduction to Software Localisation: **numbers**, **acronyms**, **e-mails**, **URLs**, **variables**, **tags**, **hot keys and shortcuts**. In Pootle, they are called **placeables** and the tool puts them in **red**, so that they can be easily recognised and inserted into the target string.
Localising NoteFly.

Click on **NoteFly 5** in the Projects section on the right of the page. Here you will find the target languages and the progress of localisation. Select **Italian** and then click on **Translate** on the top menu bar. A two-column table will be shown, with translatable strings on the left and target strings on the right. The current string will appear as a box with your cursor in the field below the source text and the options **Submit**, **Suggest** or **Needs work** on its right.

Since some strings have already been translated, you will need to activate the **Incomplete mode**, so that you can quickly move between untranslated and fuzzy strings to work on them. Go to **Filter by**, on the left bottom of the page, and select **Incomplete**, as shown in the figure below.

You can now start localising NoteFly.

⚠️ Please note that the 6th string, which is marked as **Needs work**, has to be revised.

As you might notice, the placeable `{0}` is missing in the Italian translation. Click to insert it directly in the target string. Pootle allows you to add comments to your translations. As
shown in the figure below, click on the link and add a comment to explain what you have just done and why. Then click on Submit and continue with your work.

When all the strings are translated, go to Overview on the left top menu bar and download your work by clicking on the link, as shown in the figure below.

Then Save the downloaded .po file in your folder on the desktop and Quit Pootle.
You have finished the experiment. Please, complete the Final Questionnaire.
Annex V – Virtaal Final Questionnaire

Final Questionnaire

FTI
2013-2014

USER N°: ___

FINAL QUESTIONNAIRE

- Virtaal -

Please answer the questions below concerning your work with Virtaal by ticking the appropriate box or following some other instructions. Please note that there is no right or wrong answer, so feel free to choose answers representing your opinion or to leave questions unanswered if you don’t have one.

FUNCTIONALITY

For each statement please indicate to which extent do you agree or disagree by ticking the appropriate box.

1. Virtaal does whatever I need it to do in relation to this study.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:

_______________________________________________________________________

2. There is always all the information I need on the screen.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:

_______________________________________________________________________
3. The icons and menus look like some others I have already used.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

4. If you agree, can you tell us to which other tools?
__________________________________________________________

Further comments:
_________________________________________________________________
_________________________________________________________________

5. Give a score from 1 (very poor) to 6 (excellent) to each of the following features of Virtaal’s interface.

<table>
<thead>
<tr>
<th>Features</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layout</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intuitiveness</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Familiarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Responsiveness</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. Placeables can be easily identified.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
_________________________________________________________________
_________________________________________________________________

7. The manipulation of placeables reduces the risk of typos.

☐ Completely agree
☐ Agree
8. While typing, Virtaal’s suggestions to complete words are annoying.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:

______________________________________________________________________

9. The messages displayed in the Quality Check box are always clear.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:

______________________________________________________________________

10. Give a score from 1 (very poor) to 6 (excellent) to each of the following Virtaal’s features, considering its usefulness in the localisation process.

<table>
<thead>
<tr>
<th>Features</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminology help</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
11. During the experiment, all Virtaal’s contents were visible and accessible.

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree  
☐ Disagree  
☐ Completely disagree

Further comments: ______________________________________________________________.

12. I’m satisfied with Virtaal’s reliability.

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree  
☐ Disagree  
☐ Completely disagree

Further comments: ______________________________________________________________.

**EFFICIENCY**

13. Virtaal takes too much time to get something done.

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree  
☐ Disagree  
☐ Completely disagree

Further comments:

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
______________________________________________________________________

15. Virtaal improves the localisation process.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
______________________________________________________________________

16. For me, working mostly with the keyboard is very helpful in saving time.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________

17. Considering the whole process, the time needed to localise NoteFly is acceptable.

- [ ] Completely agree
- [ ] Agree
- [ ] Slightly agree
- [ ] Slightly disagree
- [ ] Disagree
- [ ] Completely disagree

Further comments:

______________________________________________________________________

______________________________________________________________________

**USABILITY**

18. On the basis of your experiment, how difficult was it to learn Virtaal? Score from 1 to 6, knowing that 6 represents the highest level of difficulty.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

19. In your opinion, were the given instructions clear and adequate?

- [ ] Completely agree
- [ ] Agree
- [ ] Slightly agree
- [ ] Slightly disagree
- [ ] Disagree
- [ ] Completely disagree

Further comments:

______________________________________________________________________

______________________________________________________________________

20. Do you think that you would have been able to use Virtaal without instructions, eventually referring to its documentation or online help?

- [ ] Completely agree
- [ ] Agree
- [ ] Slightly agree
- [ ] Slightly disagree
- [ ] Disagree
- [ ] Completely disagree
OVERALL SATISFACTION
Please tick the appropriate box or answer the open-ended question, depending on the case.

21. In general, are you satisfied with the resulting localisation?

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
___________________________________________.

22. Did you enjoy working with Virtaal?

☐ Yes
☐ No

Further comments:
______________________________________________________________________
___________________________________________.

23. For you, what was the best thing about working with Virtaal?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
24. What annoyed you most about working with Virtaal?

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

25. In general, do you think Virtaal is a good software localisation tool?
☐ Yes
☐ No

Further comments:
________________________________________________________________________

26. Do you think Virtaal is appropriate for beginners in Software localisation?
☐ Yes
☐ No

Further comments:
________________________________________________________________________

27. What are the main advantages of this tool?

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

28. What are its main disadvantages?

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
29. What would you change?

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

Any other feedback or remark:

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

Thank you for your time!
Final Questionnaire

FTI 2013-2014

USER N°: ___

FINAL QUESTIONNAIRE
- Pootle -

Please answer the questions below concerning your work with Pootle by ticking the appropriate box or following some other instructions. Please note that there is no right or wrong answer, so feel free to choose answers representing your opinion or to leave questions unanswered if you don’t have one.

FUNCTIONALITY

For each statement please indicate to which extent do you agree or disagree by ticking the appropriate box.

30. Pootle does whatever I need it to do in relation to the translation task I have fulfilled.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
______________________________________________________________________.

31. There is always all the information I need on Pootle’s interface.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
______________________________________________________________________.
32. The icons and menus look like some others I have already used.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree
33. If you agree, can you tell us to which other tools?
__________________________________________

Further comments:
____________________________________________________________________
____________________________________________________________________

34. Give a score from 1 (very poor) to 6 (excellent) to each of the following features of Pootle’s interface.

<table>
<thead>
<tr>
<th>Features</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layout</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intuitiveness</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Familiarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Responsiveness</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

35. Placeables can be easily identified.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
____________________________________________________________________
____________________________________________________________________

36. The manipulation of placeables reduces the risk of typos.

☐ Completely agree
37. Give a score from 1 (very poor) to 6 (excellent) to each of the following Pootle’s features, considering its usefulness in the localisation process.

<table>
<thead>
<tr>
<th>Features</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminology help</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quality check</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web look-ups</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Placeable manipulation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

38. During the translation task, all Pootle’s contents were visible and accessible.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________

39. I’m satisfied with Pootle’s reliability.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree
EFFICIENCY

40. Pootle takes too much time to get something done.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
___________________________________________.

41. Pootle increases my efficiency.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
___________________________________________.

42. Pootle improves the localisation process.

☐ Completely agree
☐ Agree
☐ Slightly agree
☐ Slightly disagree
☐ Disagree
☐ Completely disagree

Further comments:
______________________________________________________________________
___________________________________________.
43. Working mostly with the keyboard helps me to save time.

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree  
☐ Disagree  
☐ Completely disagree

Further comments:
____________________________________________________________________
____________________________________________________________________

44. Considering the whole process, the time needed to localise NoteFly is acceptable.

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree  
☐ Disagree  
☐ Completely disagree

Further comments:
____________________________________________________________________
____________________________________________________________________

USABILITY

45. On the basis of your translation task, how difficult was to learn how to use Pootle?
   Score from 1 to 6, knowing that 6 represents the highest level of difficulty.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

46. In your opinion, were the given instructions clear and adequate?

☐ Completely agree  
☐ Agree  
☐ Slightly agree  
☐ Slightly disagree
Disagree
□ Completely disagree

Further comments:
___________________________________________.

47. Do you think that you would have been able to use Pootle without instructions, eventually referring to its documentation?

□ Completely agree
□ Agree
□ Slightly agree
□ Slightly disagree
□ Disagree
□ Completely disagree

Further comments:
___________________________________________.

OVERALL SATISFACTION
Please tick the appropriate box or answer the open-ended question, depending on the case.

48. In general, are you satisfied with the resulting localisation?

□ Completely agree
□ Agree
□ Slightly agree
□ Slightly disagree
□ Disagree
□ Completely disagree

Further comments:
___________________________________________.

49. Did you enjoy working with Pootle?

□ Yes
□ No

Further comments:
___________________________________________.
50. For you, what was the best thing about working with Pootle?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

51. What annoyed you most about working with Pootle?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

52. In general, do you think Pootle is a good software localisation tool?

☐ Yes
☐ No

Further comments:
______________________________________________________________________

53. Do you think Pootle is appropriate for beginners in Software localisation?

☐ Yes
☐ No

Further comments:
______________________________________________________________________
54. What are the main advantages of this tool?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

55. What are its main disadvantages?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

56. Do you think working online is an advantage or disadvantage? Please, motivate your answer.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________
57. What would you change?

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

Any other feedback or remark:

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

__________________________________________________________

Thank you for your time!