Filozofická fakulta Univerzity Palackého

Katedra anglistiky a amerikanistiky

Software Localization Case Study: Evernote

(Bachelor Thesis)

2015 Tomáš Kadlec

Filozofická fakulta Univerzity Palackého

Katedra anglistiky a amerikanistiky

Software Localization Case Study: Evernote

Případová studie lokalizace softwaru: Evernote

(Bachelor Thesis)

Author: Tomáš Kadlec

Supervisor: PhDr. Pavel Král

Olomouc 2015

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a uvedl úplný seznam citované a použité literatury.

V Olomouci dne _____ *Podpis:* _____

I would like to thank my supervisor PhDr. Pavel Král for introducing me to this interesting topic, and for providing guidance whenever I asked for it. I would also like to thank localization manager of Evernote Pavel Sapronov, for answering all my questions about the process and being available even with his busy schedule.

Table of Contents

1 Introduction

Software localization has become a widespread practice for majority of software companies when reaching global market. However, for most people it remains unknown term.

When I encountered localization during my studies, it gained my interest. Because of that, I decided it would not only be interesting to learn about localization, but also experience it first hand from the position of a translator. That is why the thesis is divided into Literature Review and Case Study.

The Literature Review will put localization in broader context of globalization and internationalization, and explain these terms together with their relationship. It will focus on the role and position of the translators and their profession in the whole process. In this fist part I will describe the reasons for localization, process and reasons that lead to localization, and then how the software must be prepared since the beginning of the development phase in order to be successfully localized. After that I will go through the theory of localization process and some trends that are emerging in translation withing localization process, as is for example crowdsourcing.

For purposed of the Case Study, I will try to localize application called Evernote. I will describe my experience as a translator in the light of knowledge gained during the Literature Overview, and compare the resources with my case. One of the goals it to show translators who might be interested in this field what to expect when entering it.

My main goal is to fully localize Evernote not only for the sake of the thesis, but also to try to have my translation used in the official release of Evernote.

In the conclusion, I will try to assess the relevancy of the reviewed literature and describe my observations of the localization process from the view of translator. I will also evaluate how close did I arrive to my goal of having the translation used in official release of Evernote.

2 Literature Review

This section has several goals. First is to to review main literature sources about localization and gather necessary knowledge for performing the localization for Case Study. Second is to introduce readers to the topic in its broader context, clarify the terms that are related to localization and introduce current trends. Third is to evaluate role and position of translation in the whole process.

2.1 Definition of Terms

The main focus of this thesis is to see localization process from the translators point of view. However, translation is one of the many parts of the whole process of localization, and even though localization industry has been growing over the years (Esselink 2000, 1), the term localization still seems like relatively unknown to general public (Dunne 2006, in Dunne 2009, 185). To clarify the terms and place translation in appropriate context, we will now look at a concept known as GILT (Pym 2004, 159), which is an acronym that stands for globalization, internationalization, localization, and translation.

2.1.1 Globalization

"It is the process by which a company breaks free of the home markets to pursue business opportunities wherever its customers may be located" (Esselink 2000, 4). It is basically the global marketing strategy of a company conceptualizing its product so it can be – after some minor revisions – sold anywhere in the world (Lingo Systems 2002, 4).

2.1.2 Internationalization

While globalization means creating a strategy that does not have to be tied to one product (software), internationalization is the phase that concerns specific product and its creation. Internationalization takes place during the product development cycle (Esselink 2000, 2). It includes extracting text from software, so translators cannot break the code, or implementing ability to support character sets for particular languages (Esselink 2000, 3). Simply put, the key tasks of internationalization is to ensure that it will be possible to localize and translate the product (Esselink 2000, 25). Impacts on translation will be further discussed in chapter 2.3.

2.1.3 Localization

"The process of customizing a product for consumers in a target market so that when they use it, they form the impression that it was designed by a native of their own country" (Lingo Systems 2002, 4). The idea of perfect localization is creating a product about which user would not be able to tell if it was was created in his country and his language, or if is foreign product. That is why localization is not simple translation of the text of particular software, but also adapting it to be culturally appropriate (Esselink 2000, 3).

2.1.4 Translation

Lingo Systems (2002, 4) describe translation as "the process of actually converting the written word of a source language into the written word of a target language." It is worth noting that in translation within localization there is usually considerable potential for using CAT tools. It is due to need of terminology managers or translation memory, because there is generally high amount of repetition in software texts (Lingo Systems 2002, 33). Further discussion on the translation within localization process will be reviewed in chapter 2.4.2.

2.1.5 Summary

Quote from Anthony Pym (2004, 30) is very helpful in summarizing the GILT:

> We might say that there is one wide process called "globalization", of which "internationalization" and "localization" are parts. In order to globalize, you first make your product in some way general ("internationalization"), then you adapt ("localize") to specific target markets ("locales").

It is also important to realize that even though these terms are in direct relationship with the translation, they originally come from marketing. and their primary concern is working with product, not just text and linguistic constructs (Pym, 2001).

2.2 Reasons for Localization

The main reason for localization is to introduce the software product to markets in other countries. In case of software, this process is especially simple due to a low-cost worldwide sales platform (LISA 2003, 8-9) that is provided by Internet. In business sphere, this is naturally motivated by increased market potential, so we will now look at the two main questions concerning this issue.

2.2.1 What is the demand for localized software?

The common belief among people, especially young generation, seems to be that localization is obscure, because *nowadays everybody speaks English*. However, research among users shows this is simply not true. Common Sense Advisory conducted study about this issue:

> "Based on a survey of more than 3,000 global consumers in 10 non-Anglophone countries in Europe, Asia, and South America, 75% prefer to buy products in their native language. In addition, 60% rarely or never buy from English-only websites ... Our research in 2006 proved that 72.4% of consumers surveyed were more likely to buy products in their native language. Our 2014, larger-scale behavioral study of consumers again validates this preference and, in fact, concludes this demand is increasing, with a full 75% of respondents saying they want the products in their native language".

These data prove that there is significant demand for localized products. Consumers want to buy product in their language.

2.2.2 Will the investment be worth it?

While it is clear that localization significantly increases market potential of the product, and there is high demand among the users for localized software, it is not so clear what is the localization Return of Investment (ROI). In its *The Localization Industry Primer*, LISA[1] states that "20 largest IT companies alone are leveraging around USD 1,5 million year to generate sales of some $15 billion, an incredible ROI of over 1000% (LISA, 2003)", which is $10 return on every dollar invested. Speaklike quotes these same data from 2000 in its report on localization and adds newer study from LISA conducted in 2007, which claims that average return on the dollar increased to $25 (Speaklike 2010).

However, other sources bring more realistic insights in these highly optimistic statistics. In his article for *Software Business*, Benjamin Sargent also quotes the LISA study from 2007, but argues that "there are other costs associated with foreign markets that are not factored into that equation that cannot be ignored" (Sargent, 1). He also argues that there are other factors to be considered, such as whether the product is already established on the market without localization, since localizing established product yields better cost benefits.

Sargent's argument about other costs and need of additional research is further underlined by the fact that not many companies are able to get clear data on

1 *Localization Industry Standards Association*, which shut down on 28 February 20011.

localization ROI. In DePalma's survey from 2005, only 26% of interviewed companies “said that they could formally measure and calculate the return on their localization investment” (DePalma 2006, 18). Most of the companies never went back to reevaluate whether the localization was a good investment and should be continued, or if it should be shut down before more money is wasted (DePalma 2006, 19). More generally speaking, 74% of DePalma's respondents answered that they had a much less concrete sense of what their investment yields, often citing localization as “just a cost of being a global business” (DePalma 2006, 19). I believe this paints more realistic picture of state of affairs in the industry.

2.2.3 Conclusion

Translation is directly affected by the business. If there was no demand for translation and no profit to be gained by the localization, the motivation of companies to hire translators for localization process would be significantly decreased. It is also interesting that while this is the case, large number of companies do not implement any mechanics for assessing how much the investment into localization pays back, which puts translators in peculiar position: companies realize the demand, but they are not sure what this demand brings economically to their business.

2.3 Internationalization: Preparing Software for Translation

Once it is decided that a company wants people from other countries with different languages to use its software, the software must undergo several steps in order to enable localization and translation.

It was mentioned in chapter 2.1.2 that internationalization serves as preparation of the product for localization. This means that the quality and extent of internationalization has significant impact on translation. Poor internationalization can cause many problems for the translators, force them to undertake unnecessary measures, complicate the process of testing, or even make the translation impossible.

We will now look at key issues that need to be addressed by the developer, and what these issues mean for the translator. In the chapter on Case Study (3.3) I will also describe which of the following issues I encountered during my localization experience and how did they affect my translation process.

2.3.1 Extracting text from source code

Main object of internationalization is to extract text, that is going to be displayed to the user, from the source code of the application (Lingo Systems 2004, 42; Dunne 2009, 191). That way the translator, who often does not understand software engineering, cannot break the code (Lingo Systems 2004, 44). Separated text is then stored in resource files that are used by the software (Dunne 2009, 191).

There are several ways of achieving that, and we will briefly summarize the most common ones, as they influence the translation process.

Lingo Systems describe these three approaches (2004, 54-55).

1) Using third party software that lets the linguist work through the User Interface components and translate the parts that need translating. These tools often provide graphical editors for adjusting dialog boxes or buttons. Great advantage of this approach is that the translator can see the text in of the application context. However, additional skills and expertise are required for working with such tool.

2) Resource files contain the source code, but it cannot be edited. Translator cannot break the code, but it can provide much needed context. Of course the context is useful only if the linguist can read and understand the code.

3) Text is extracted from the resource files and placed in a word processing document with pointers to their original placement. Once the text is translated, it is reinserted into the resource files. Main disadvantage of this approach is that the translator cannot see the text in the context of the application. While this is significant problem, this approach is widely used because of its simplicity and lowest demand on translators software engineering skills. Microsoft (2015)[2] calls it “the simplest and most straightforward method”, and adds that another advantage is the simplicity of removing content that is no longer used by the software. That is important because translators will not be able to tell which content is in use and which is redundant.

2.3.2 Allowing for text expansion

All buttons, tables and dialogue boxes should have approximately 30% extra space, because translated text is usually longer than English source text. In some cases it can be much more than 30%, and Dunne (2009, 211) argues the expansion can increase by more than 300%. Resizing the User Interface later is not cost and

2 Chapter *Isolate Localizable Resources.*

time effective, therefore it is not advised to design User Interface in the source language with significant space restrictions (Esselink 2000, 33). Lack of space complicates the work of translators, who are forced to come up with shorter translations, or if it is not possible, to request developers for adjustment of space, which can take significant amount of time.

2.3.3 Separating text from graphics and choosing appropriate images and colours

There are two issues concerning images, colors, and icons.

First issue is text within any kind of graphics. For example text that is part of a picture is hard to extract, and it can mean significant challenge for the translator. Even in the case of the translator being able to edit the graphics, it adds to the time and cost (Esselink 2000, 27). It is therefore advised to keep the text in graphics to minimum, (Ibid.) or create the text as a separate component (Lingo Systems 2004, 23).

Second issue is more general. Some colours or their combinations can be received differently in various cultures. Esselink advises to avoid using colours associated with national flags or political movements. He also points out that icons and other pictures should not contain body parts, holiday symbols, or body language (2000, 34). Typical example would be raised thumb, which can signal approval or success in some cultures, but it is sign of significant rudeness in others. The general advice is to use "color schemes and graphic selection that avoids offending potential customers" (Lingo Systems 2004, 6). However, this is easier said than done. Software designers cannot comprehend all the target cultures and take them into account. That is why it is advised to ask localization team to provide feedback: it is better to have unified graphics, because designing several graphic variations can be expensive, and also confuse users who would be working with both localized and original versions of the program (Microsoft 2015)[3]. Translators working on localization are often expected to point out such issues, should they encounter them. However, if the whole colour scheme or most icons would be problematic in the given culture, the translator would typically not be able to solve this problem and he would have to wait for developers to adjust the software's visuals.

3 Chapter *Localization Elements*.

2.3.4 Enabling Editable Keyboard Shortcuts

Keyboard shortcuts (hotkeys) are available in most applications and operating systems. As an example, we can name perhaps the most widely used combinations: *Ctrl+C* and *Ctrl+V* for *Copy* and *Paste*. As this example demonstrates, keyboard shortcut typically consists of *Ctrl* key and specific letter on the keyboard (Dunne 2009, 194).

Although most of the time keyboard shortcut and hotkey are synonyms, Dunne (2009, 194) makes distinction between the two. He calls the Ctrl+ letter accelerators, and hotkeys defines as "application-specific shortcuts that enable the user to access commands in menus or in dialog boxes by simultaneously pressing the left-hand Alt key on the keyboard plus a specific letter." Similar distinction is made by Esselink (2000, 71-72).

Keyboard commands should not be hardcoded (it should be possible for the translator to change the key combination for the function), because they need to be adjusted for specific language (Esselink 2000, 4). Sometimes the motivation for the change can be linguistic, as the first letter of the translated word for the given function does not reflect the letter used in the shortcut (Lingo Systems 2004, 47). However, many times the shortcuts are already established by other localized products and it it is recommended to use those (Microsoft 2015)[4]. Dunne (2009, 194) points out that if the application is running on Windows, shortcuts should be consistent with the shortcuts used in the specific version of this operating system.

Hotkeys are typically subject of more changes based on translation than accelerators, because they serve as navigation within the menu using letters, which often correspond with the first letter of the word, and translation typically changes that letter. If the translator decides to change a hotkey, it is important to not create conflict by assigning one that is already in use (Dunne 2009,195).

2.3.5 Choosing Appropriate Language and Style

The text needs to be understood in different cultures, thus the style should be basic and uniform. That is typically achieved with help of short sentences, simple vocabulary, consistency, and careful punctuation. There should be no culturally specific jargon, humour and slang. If possible, abbreviations should be avoided because it is difficult to reproduce them. Long modifier chains should not be used

4 Chapter *Localization Elements*.

(“required disk slot” can mean both ¨”the slot for a required disk” and “disk slot which is required”). The general aim should be to produce sentences and information that can be interpreted only one way (Esselink 2000, 27-28). If such language and style is achieved, it significantly helps the translator. As was described in section on extraction of the text (2.3.1), the translators often do not see the context of what they are translating, thus expressions with only one meaning are essential.

2.3.6 Variables and Composite Strings[5]

“When creating strings for output, well-intentioned programmers use a coding trick that has been passed down from generation to generation of developers as a good coding practice. That trick is using text strings with variables that get composed at the application's run time” (Microsoft 2015)[6]. This practice is very common. Good example are numbers: instead of creating variations for sentence “Are you sure you want to delete 2 selected files?” containing every possible number combination, the developer can insert some kind of variable (we can use X for purpose of this example), so the sentence looks like this: “Are you sure you want to delete X selected files?”. The application then replaces X with appropriate number of files that has been selected.

While this trick is a great utility for the developer, it is usually nightmare for the translator (Microsoft 2015)[7].

The variables for numbers are not particularly hard to determine, but the problem does not end with that. In case of a sentence following a number, English needs only two variations of the given sentence: one for singular and one for plural case (“1 selected file” and “2 and more selected files”). This is not the case for many other languages (including Czech), which can have three or more variations of the word *file* depending on the number it is referring to (Dunne 2009, 205).

Even bigger issue are strings that are composed from two sentences. In that case, the translator has no way of knowing gender of the noun in the place of the variable (Lingo Systems 2004, 47).

Microsoft (2015)[8] advises not using composite strings with variables, but at the same time admits that there are times when it is necessary. It is however strongly

5 String is general expression for any set of characters.
6 Chapter *String Handling.*
7 Chapter *String Handling*.
8 Chapter *String Handling.*

suggested not to compound several variables, because it is very difficult for the translator to comprehend the issue and many times impossible to provide translations that would keep correct word form in the given language in all the combinations possible.

2.3.7 Enabling Operating System Support

It is important that the translator is be able to use all the special characters of the target language. These special characters can create issues with sort orders. For example in Czech, letter "a*"* is alphabetically followed by its accented version "á". However, most Scandinavian languages put their accented characters after the letter *"*z*"* (Lingo Systems 2004, 41).

Another issue are formats of numbers, time, currency and date. Format of these elements should not be fixed, because different countries use different formats (Esselink 2000, 26).

Developers usually do not have to come up with their own solutions to any of above problems, since "using standard APIs today (such as those provided by Microsoft), handles much of the conversion" (Lingo Systems 2004, 45). This means that developers can use the globalization infrastructure that is part of Windows (for more details see Microsoft 2015, chapter *Use Locale Model*). If such measures are implemented, translators do not have to worry about transferring any of the above, as it should be done automatically by the Windows Locale Model. However, if the measures are not in place, there is very little the translator can do to implement these cultural specifics.

2.3.8 Conclusion

Translators are not directly part of internationalization process, and since this process is often performed during the development phase of the software, there is very little translators can do to influence it. However, quality and extent of internationalization determines the potential for appropriate translation. This puts translators into unfair position: they cannot influence the process which directly determines quality of localization, and at the same time they can be often subject of blame when the translation is inappropriate because of poor internationalization.

2.4 Localization Process

Once the internationalization is complete, localization process can begin. This chapter will focus on specifics of translation in localization process, and we will discuss in more detail the relationship between the two.

2.4.1 Steps

Summary of typical steps of localization by Dunne (2009, 192) is helpful for our purposes:

1) Receipt of source language files.

2) Translation of all translatable strings. Usually some CAT or localization tool is used.

3) Linguistic testing – spelling, consistency, accuracy.

4) Cosmetic testing – correcting errors in the visual aspects of user interface

5) Functional testing – testing the running application for linguistic, cosmetic and functional defects.

We will now look at this areas in more detail.

2.4.2 Translation

Translators receive the resource files with content intended for localization (or some kind of access to them) and translate them. During the translation they have to deal with many issues described in chapter on section internationalization rocess (2.3): strings out of context, composite strings containing variables, different keyboard layout, and hotkeys. Frequency and difficulty of these issues depends mainly on the quality and extent of internationalization.

Description of my encounters and solutions with these issues is available in section 3.3. In the rest of this section we will briefly look at the relationship between current translation theory and localization.

One of the definitions of localization is, that if the product is well localized, the users will not be able to tell whether the it was created in their own language, or if it was localized (2.1.3). Since this is the same direction the translation theory has been following last twenty years, one might assume that translation theory and translation within localization are compatible. Interestingly enough, this is typically not the case:

> Studies have allowed us to see interpersonal dynamics and cultural

> specificity as playing major roles in the solving of translation problems. And now, precisely when some translation theorists are on the point of affirming that all crosscultural relations are translational, that translators should be experts in the management of cultural difference and the like, right at this point of maximum expansion, translation theory is being outflanked by the discourses of localization and translation itself has been returned to the narrow linguistic exercise it was in the 1960s. (Pym 2004, 52)

Pym argues that translation is no longer regarded as a sentence-leveled language replacement exercise, but translation within localization is often exactly that. As was shown in chapter 2.3, the text is typically extracted and placed outside of context, divided into segments in length of words or sentences, and translated in such format. Translators influence the cultural differences in localization, but those are mostly outside the text, in form of icons (2.3.3) or hotkeys (2.3.4). In regard to text, there is very little room for creativity.

2.4.3 Testing and Quality Assurance

After the resource files are translated, they are inserted back into the software. After that, translators (and in some cases software developers) begin to test all parts of the software for several potential problems. Testing phase is divided into the categories based on what sort of problems are being dealt with.

Linguistic testing: It "focuses on all language-related aspects of a localized application" (Esselink 2000, 150). Most of the translators will have spell check available in the environment they use for translation of strings, which can significantly help with typos. However, there will be many problems concerning the form of words influenced by context, and computers currently cannot fully process such complexities.

Even this first stage of test should be carried out in running application. For example the composite strings in dialog boxes or strings with variables cannot be typically reviewed elsewhere. There can be issues like mistaking separated verb for a noun, but in context it will be clear which is which.

Visual/cosmetic testing: Localized version must display same number of user controls like menu options or dialog boxes. The accented, extended and special characters must display correctly (Esselink 2000, 151-152).

The primary focus is to ensure that the text expansion (as described in 2.3.2) did

not create strings which do not fit in borders of their placement (Dunne 2009, 211).

Functional testing: The software needs to be tested for function, preferably on computer with native operating system. If there were changes made to the source code for the reasons of localization, such parts should receive special attention (Lingo Systems 2004, 61). The primary goal is to ensure the same functionality the software had before localization (Dunne 2009, 212) and it "usually mirrors the testing process that has been performed on the source language product" (Esselink 2000, 152). Generally, the translation should be final and stable, and the focus is on compatibility with various locale hardware and software, like operating systems. This stage is not exclusive to translators – software engineers often participate too (Ibid). For the translators this might not seem so different from the previous rounds of testing, and Dunne confirms that this is often the case: "functional testing also often serves as another round of linguistic and cosmetic testing" (2009, 212).

Quality assurance: While all the above criteria are predominantly objective and can be measured, the outcome quality is a matter of subjective perception of the user. As Dune puts it: "In today's market, quality is whatever the client says it is" (2009, 218). This makes quality testing before the release of software purely theoretical discipline, because (aside from objective parts mentioned in testing) the end users will not assess the quality using expertise and necessary knowledge, (the user for example cannot know that compromise in translation had to be made due to limited space) but rather their own criteria, which differ from user to user. Of course there is objective linguistic criteria for evaluating the translation and cultural norms to evaluate the localization, but from the marketing point of view, another important set of criteria is the reception of the user. Dunne expresses this issue quite brilliantly:

> Assessing the quality of a localized product on the basis of the subjective expectations and/or preferences of a reviewer, rather than on the basis of formally specified requirements, is akin not merely to changing the rules in the middle of the game, but rather to allowing the rules of the game to be changed by each new player who enters the playing field. (Dunne 2009, 220)

Dunne then proposes to assess needs of users and clients before the start of localization process, but the decision about localizing into majority of languages is often made after the release of the product, which makes objective, empirical

quality assurance impossible in these cases (2009, 218).

2.4.4 In-house or Outsourced Translations

When we are considering translation in the localization process, important issue for the company is to decide what model is it going to use for hiring translators. This chapter is not focuses on viewpoint of a translator, but rather on how the company can work with translators during localization process. Golota in her article at *Globalme* argues that there is no clear answer: "Everything depends on your business model, and the volume of localization work you anticipate." She then goes on to list some advantages and disadvantages of both options, which I will now summarize.

In-house translation is more suitable for continuous and frequent updates. It provides direct contact with the translators, which enables them to react faster to changing schedule or immediate requests. The fact that translators are part of the company also usually means that they know the product much better than outsourced teams. On the other hand, the initial cost of establishing such team of translators within the company will not be insignificant, and the extra costs do not end here: the translators will require licensing and continuous training. Another disadvantage is that the in-house team will be limited compared to outsourced company with more people, which represent more expertise, more native speakers of more languages, etc... This last point is also mentioned by Lingo Systems (2004, 13).

The main advantage of outsourced localization is the cost: company pays only for the number of words (or whichever unit is chosen) that had been translated. Having in-house team means paying translators even if there is no work for them at the moment.

Nielsen presents in her article for server CodexGlobal third option: partly outsourced localization model. This model involves outsourcing only part of the localization, usually the translation and proofreading. This model seems like an attempt to combine advantages of both models while avoiding the disadvantages.

Authors of these two articles talk about the outsourcing in terms of agency which takes over the whole localization process, or at least part of it. Tomala (2013, 36) writes in her article for *Core Focus* by *Multilingual* about differences in working with freelancers and agencies. She argues that advantages of working with

freelancers are direct contact with the translator and higher level of flexibility, but acknowledges that working on bigger projects with freelancers can get very complex and difficult to manage. At the end of this passage, I would like to quote her comment on working with freelancers, which seems to make the subject more humane, but at the same time it reflects my experience: “It may seem that vendor management is a never-ending chat with translators and reviewers, including sharing family photos and holiday postcards” (Tomala, 2013. 35).

2.4.5 Crowdsourcing

Crowdsourcing is on rise both as general phenomenon and as translation method, and it is actually the method I was part of in my case of localization. The whole discussion on the topic, its success stories, and dangers is quite complex, so we will focus on those parts that are connected to translation and influence the translation market.

“The term ‘crowdsourcing’ was coined by Jeff Howe as a portmanteau of ‘crowd’ and ‘outsourcing’ in 2006” (European Commission 2012, 9). Server *Daily Crowdsource* provides clear definition in it's article *What is Crowdsourcing*: “Crowdsourcing is the process of getting work or funding, usually online, from a crowd of people... The idea is to take work and outsource it to a crowd of workers.” The main idea is to utilize the help of volunteers through various platforms instead of hiring professionals. What the amateurs lack in time and training they make up in numbers.

Most famous examples of success stories are Linux and Wikipedia, both using crowd of amateurs to achieve great, ongoing, and deep impact in their field. Maybe that is why *Daily Crowdsource* is very optimistic about the advantages of this method. The mentioned article argues that results achieved by crowdsourcing are faster and client can choose from best ideas of many people, instead of getting just one best idea of singe professional. While this is certainly true in case of manual tasks as for example tagging pictures on web, in areas like translation the need for professional seems to remain.

Criticism of crowdsourced translation is usually twofold. First objection is the mentioned need for professional training and experience in the area of translation: “The reliability and value of crowdsourced work is often questioned, compared to work carried out by skilled and knowledgeable professionals” (European

Commission 2012, 44). Second is concern for the status of professional translators: should this practice become established, it could lead to disruption of market, and many professional translators being out of work because they cannot compete with the "price" of free translators (European Commission 2012, 23). This disadvantage for translators is usually the main reason for companies to use crowdsourcing: nothing is more compelling for business as free resources. The issue here is not only that the volunteers are lowering the value of translation for professionals, but also that volunteer crowdsourcing for Linux or Wikipedia creates products that are available for free to anyone everyone, meanwhile "businesses can easily exploit crowdsourcers and make undue profit from the work the latter are happy to carry out for free or at very low rates" (European Commission 2012, 45). Many translators fully understand this issue and are not happy about it. Probably the most famous manifestation of this phenomenon is the case of LinkedIn. The server tried to follow the steps of crowdsourced localization of Facebook and Twitter, and asked professional translators to localize LinkedIn for free (Common Sense Advisory 2009). Translators found it quite offensive and even formed group "Translators again Crowdsourcing by Commercial Businesses" (LinkedIn 2015).

While crowdsourcing carries this potential for abuse by commercial business or causing disruption to the market, very few people argue against the advantages it presents for the non-profit organizations or free services.

2.4.6 Never-ending story: Continuous Localization

Until now, localization was perceived in this thesis mainly as one time process: company decides that the product should be sold on global market, the product is internationalized and then localized. Product is then tested and sold, so one might presume that this is the end of the whole process. However, increasing number of current applications is regularly updated, some in fairly short intervals. Users naturally want to to have access to the latest version and its functions. "Software development never stops when the first version of a product is released. That is only the start of a new phase in development: bug fixes, minor updates and at some point major new versions will be developed and released – continuously" (Getlocalization 2015). This is not the case of every software, but it is becoming more and more of a trend. I used to have my *Gmail* account switched to English,

simply because the new features like chat were first available in English, typically weeks before they were implemented in Czech version. If the software is prepared for continuous localization, this process can be significantly faster. “Continuous localization integrates localization into the product development process, meaning translations happen concurrently with development” (Intel Developer Zone 2013). This requires integrating platforms that will take updated strings of code, prepares them for localization and notifies the translators about new content. For translators, this means that after the main work of translating some product during the localization phase, there is possibility of contract for additional and continual services for the updates.

2.4.7 Conclusion

Translators are directly part of the localization process. They receive the text, translate it and test it. However, the translation within localization has several specifics and it does not leave much space to the translators creativity. Translator is translating words and sentences. There is no context and the main focus is on basic equality, typically on level of words and sentences.

3 Case Study

The first part was aimed at gaining sufficient knowledge for performing localization. In this part, I will describe my experience with localization and compare it to the first part of the thesis, which should help to frame the case study in appropriate terms and also who the relevancy of reviewed sources.

However, the main goal is to experience the localization process as translator and hopefully complete the localization to the extent where it could be used as official release.

3.1 Choosing the Software

First idea was to find some open-source software and localize it. The main reason for this option was the license issues – localizing any other software without permission would be a violation of copyright. I was searching through servers that collect free and open-source apps and asking other people what software they use for everyday tasks, when I learned about Evernote. At first it seemed like it is not what I was looking for, because even thought basic functions of Evernote are free, it is not freeware and definitely not open-source. Before I discarded this idea, I noticed “Help us translate” button on the Evernote's web. It forwarded me to translation server for which I was required to create an account. With existing account I accessed the translation server running on Pootle (Pootle will be explained in following section 3.2.1) and found out that anyone can suggest translations.

There was also possibility to contact localization team. I wrote an email explaining my situation and offered to localize Windows client into Czech. In return I needed exported strings or preferably localized client for the purpose of my thesis. I received very quick response from localization manager Pavel Sapronov, who offered to consult this issue over Skype. After short talk we agreed on aforementioned terms. Naturally there was no promise of whether the client would be released and the manager could not decide what to do with the localization at this point, but I was promised to receive localized version of the client for the thesis, which was satisfactory for my purpose.

3.2 What is Evernote

There will be significant number of references to Evernote in the rest of the case

study, so in the following section I will summarize some main functions of the application.

“Evernote is a platform for human memory, designed to help individuals remember everything” (Evernote, 2010). Evernote is designed to acquire and store information. It supports storing text, audio, video, screenshots, scans, etc. Some advanced functions of Evernote can further process the scans (adding social media information to scanned business card, optical character recognition). All this information can be stored to *Notes*. Two or more *Notes* compose a *Notebook*, and two or more *Notebooks* can create *Notebook stack.* One of the key features of Evernote is accessibility of user's data: the application supports several versions of Windows, iOS, Android and more. All the stored information can be synchronized: if a user stores some note on desktop computer with Windows that is connected to internet, that data will get uploaded to the online storage. The user can then for example run Android version of the app on his tablet and via synchronization all the data created on the desktop will now be downloaded to the tablet. Synchronization is limited by upload allowance, which depends on account level. All the *Notes* and *Notebooks* can be explored with full-text search, making the stored information available on all devices all the time. Evernote has several other functions, among which are *Annotation* (marking up and editing images like screenshots) or *Presentation* (simple presentation of the notes without the need of creating slides).

There are three levels of accounts. Evernote Basic provides basic data storing and editing functions for free. Additional functions and online storage room can be acquired by subscription to Evernote Plus, and unlimited storage and most exclusive functions are part or Evernote Premium.

3.2.1 Evernote Translation Platform

There will also be significant number of references to the platform Evernote uses for translation, so even though this is technically part of the translation section (3.3.8), it is placed here for better comprehensibility of the following information.

Evernote uses tool called Pootle, which is translation server with many translation features. Pootle is web application that allows working with several tools (e.g. API Translation Toolkit). It arranges the order of content assigned for translation into folders according to different languages and projects, provides Translation

Memory (TM), and supports crowdsourcing translation (Zajiček 2013, 23).

Evernote uses its own modified version of Pootle. Anyone with Evernote account can log in, choose project and language, and start working on translation. Until determined otherwise, the user can only s*uggest* translation. There is possibility to use Machine Translation (Google API), copy the original text to the translation segment (useful for variables, as it significantly lowers the possibility of misspelling longer variables), or use translation from Translation Memory and/or translations suggested by other users.

Translation Memory is in form of L10N Robot, which is automated Robot with several functions: it (1) stores all the translations and suggestions, (2) automatically translates when there is 100% match with Translation Memory, (3) translates when there is need for only slight adjustment (e.g. if there is translation of the word *Note* as *Poznámka* in TM, and L10N Robot encounters word *note*, it will take the unit from TM and change the first letter to lowercase, resulting in *poznámka).*Possibly the most important feature of Pootle is that if there is part required for the application to run properly, (variable, shortcut) it has to remain the same in the translated text, otherwise the Pootle will not save the translation. This eliminates any potential of errors caused by translator, who could omit one of the variables or break the assigned shortcut.

3.3 Localization of Evernote

In section 2.3 it was described that before software can be localized, it has to undergo the process of internationalization. This section will have very similar structure as sections 2.3 and 2.4. It fill follow the issues described in these sections and comment I will describe my experience with them during the case study.

As was argued in section 2.3, internationalization directly affects translators options in the translation and cultural adjustment process. Therefore, I will assess how well was Evernote internationalized, and how it specifically affected the translation I performed.

After that we will go through the process of localization and my experience with it, as was described in section 2.4.

3.3.1 Extraction of Text from Source Code (2.3.1)

The whole Windows client and its functions have it's text extracted and placed

under several different folders in Pootle. This corresponds with the third mentioned approach from 2.3.1 – the text is placed in a word processing document with pointers to their original placement. This means that I as translator have no access to the source code, and have to deal only with several variables or keyboard shortcuts. It was comforting that I could not break the code of the application even if I gave it my best attempt, but as for the aforementioned disadvantages, I was translating content with very limited context. Only help with that was that every part of the application has its own folder and usually several sub-folders (Figure 1).

Figure 1 shows different folders for different parts of Evernote text. Word count is in the column headlined *total.*

This means following: since functions like *Annotation* (js-markup on Figure 1) or Presentation (js-present on Figure 1) have their own folders, I know that when I encounter strange phrase (e.g. *Blue Laser Pointer*) or generic word (e.g. *View*) in folder js-present, I can run the *Presentation* function of Evernote and look for the given text. While this can be of great help, its potential is limited by some very large folders. It can be seen on Figure 1 that all the folders have maximum of several hundred words, while *enclient* (folder for the actual Evernote client) has 12779 words. It is further split into several other sub-folders, but we can see on Figure 2 that same phenomena applies: the client (Evernote.rc) and message boxes have by far the highest word count.

Figure 2 shows that only two folders have majority of content. Word count is in the column headlined *total.*

The client folder Evernote.rc contains no additional sub-folders (Figure 3).
This means translating 5706 words that could placed anywhere in the client and that often repeat (there is roughly 10 instances of the word "View" via full-text search, as it can be used both as noun and verb), and there is very little context. It can be presumed that the word View in the chain of words like *File*, *Edit*, *Format* and *Tools* is part of the *Main Menu*, but there are no actual guarantees of that. The most efficient option seemed like focusing on context in testing phase, since it is not hard to generate updated localized version.

Figure 3 shows folder that has no additional sub-folders. Several statistics are available. Clicking on *View all* will allow fo viewing and modifing the actual strings.

One example can be seen in Figure 4 in a list of Premium Features (*Funkce Premium*). Because of lack of context, I evaluated most of the phrases (*Search inside Attachments, Turn Notes into Presentations*...) as commands that will let the user execute them (thus by clicking, user would start searching inside attachments).

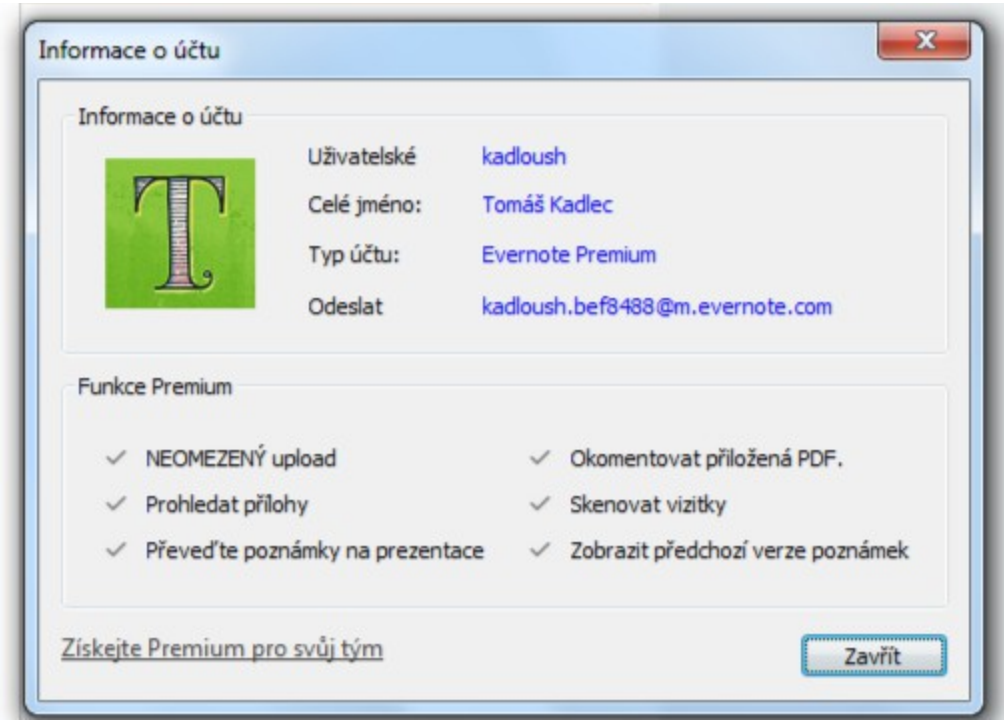

Figure 4 shows wrong translation caused by lack of context.

The correct interpretation is that they are list of features, so instead of commands as seen on Figure 4, I changed it into list of features (Figure 5). That corrected the linguistic mistake. The remaining issues of space visible on the Figures 4 and 5 will be addressed in next section (2.3.2).

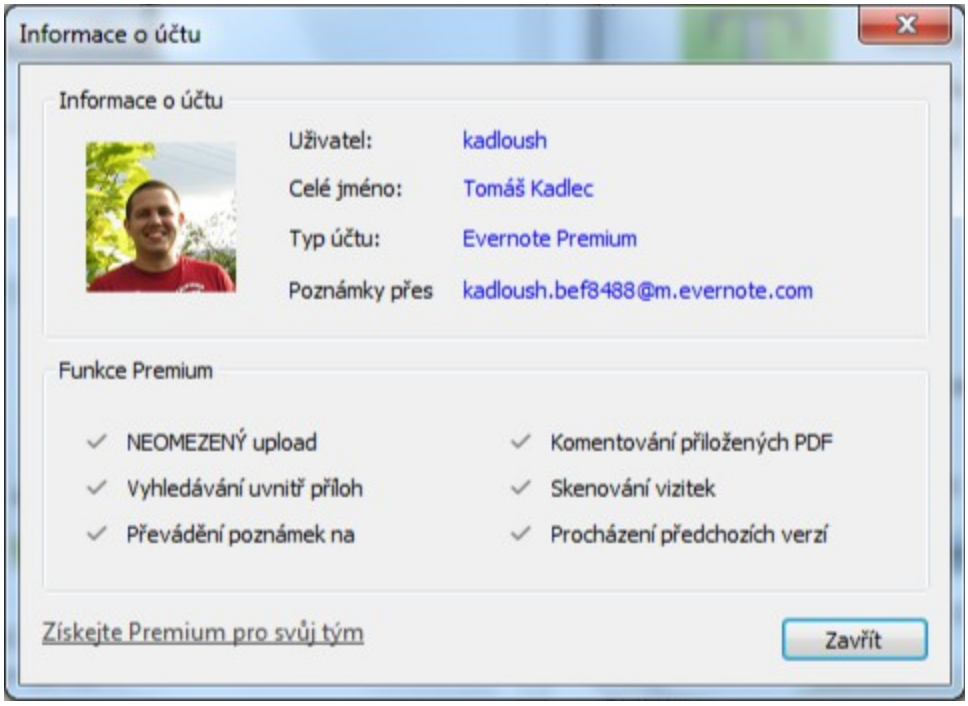

Figure 5 shows correct translation.

However, not all the text that displays to the user was extracted for translation. There are several dialog boxes or pop-up windows that were in English after the translation of all strings in Pootle.

Also, there is significant number of redundant strings. The text was extracted from previous version of software, inserted in Pootle, but after the text was no longer needed because of update, it was not deleted. It became obvious when I encountered string that mentioned previous version of Evernote, as seen on Figure 6. I provided a translation because untranslated strings in Pootle will disallow marking the project as finished, which will disable adding new language to the application, but the translation will not be displayed anywhere. Evernote localization manager estimates that at least 5% of strings in Pootle are redundant.

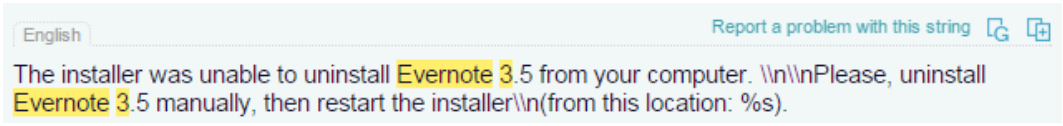

Figure 6 shows redundant string

Apart from strings that are not used in the software anymore, I encountered several strings that were internal, so only the software developers can see them. There is no point in translating these, but it was impossible for me to tell with certainty whether the strings are for internal use only, so they had to be translated anyway.

3.3.2 Allowing for Text Expansion (2.3.2)

Vast majority of the application had enough of additional space needed for Czech language. Expanded translations like *Nová poznámka* (14 characters) fit everywhere even though the original phrase *New Note* has only 8 characters (over 60 % increase).

However, there were few instances where I had to come up with different solution for the translation because of space restriction, and even one case where I did not manage to shorten the translation enough to fit.

First case was the word *Username* (8 characters) in the Account Information sub-menu. Common translation is *Uživatelské jméno,* which has 17 characters, making the increase just a little over 100%. As can be seen on Figure 4, there was not enough space for that, so I shortened the translation to *Uživatel* (User). I believe it conveys the message. Additionally, the actual username is displayed right next to it, so there should be no misunderstandings (result can be seen on Figure 5).

Second case was phrase *Email notes to.* My first version of translation was *Odeslat poznámky na:* but as can be seen on Figure 4, only the first word was displayed (not mentioning that the translation did not convey the message, but we will discuss that more in section 3.3.5). I attempted to shorten it and correct it at the same time, coming up with awkward *Poznámky přes:* (Notes via), shortening the original translation from 20 to 14 characters. The translation was not only very inaccurate and confusing, but also the ":" still did not fit. This whole issue resulted into consultation with the localization manager, who reacted on the issue with words: "Oh, that happens all the time," confirming this issue as one of the common problems of localization from English. After examining this menu, he

decided to hand this issue to developers for readjustments, especially since Czech was not by far the only language facing space restriction problems in this particular setting. In section 2.3.2 it was argued that this kind of issue can take considerable time to resolve. It is estimated by the developers that the issue will be fixed by the end of May, which is roughly 6 weeks after it was reported. Since it is fairly significant issue of several languages, it seems like serious deficiency in the internationalization process.

3.3.3 Separating text from graphics and choosing appropriate images and colours (2.3.3)

First issue described in section (2.3.3) was concern for text separation from the graphics. Evernote is well internationalized in this area: when there is text in a picture or bitmap background, it is separated and available in Pootle. After the translation, the text is placed back. We can see example of one result in Figure 7.

Figure 7 shows text, which was extracted from the bitmap background, translated and automatically inserted back, which is extremely helpful for the translation process.

Second concern of section 2.3.3 was for colour combinations or icons that are culturally transferable. I did not encounter any offensive or misleading icons, nor were the colour combinations inappropriate for Czech culture. This may be partly given by the fact that Evernote is mostly displaying text. However, the icons used for functions are simple and clear.
We can see examples in Figure 8 and Figure 9.

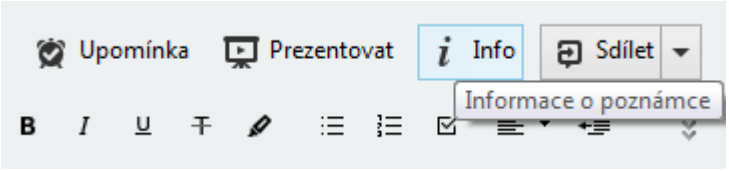

Figure 8: From left to right – Reminder, Present, Note Info, Share.

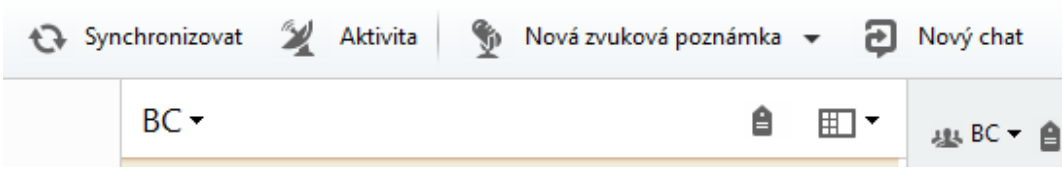

Figure 9: From left to righ – Sync, Activity, New Audio Note, New Chat.

In Figure 8 we see *Synchronization, activity, New Audio Note* and *New Chat.* Microphone for audio is very accurate, and Chat and Synchronization do not seem to be creating any confusion. The satellite for *Activity* might be possibly misleading, but simple click showing the recent activity eliminates any rooms for wrong understanding of the concept. In Figure 9 we see alarm clock for *Reminder* and movie screen for *Presentation.* Since the names of the functions are quite clear, the icons seem to be mostly complementary, nevertheless quite accurate for Czech setting. One might question the fact that there is same icon for *New Chat* in Figure 8 and *Share* in Figure 9. That is because the default function of *Share* (if not specified otherwise by the downward facing arrow) is to *Share in Chat*.

3.3.4 Enabling Editable Keyboard Shortcuts (2.3.4)

There is vast number of keyboard shortcuts available in Evernote. We will now look at two types of keyboard shortcuts, as we distinguished them in section 2.3.4. Both kinds of Evernote shortcuts are not hardcoded and the key combinations can be changed, as there might be several reasons for changing them.

First group we established in 2.3.4 were accelerators – combination of Ctrl key and some other key/keys. In Evernote, these are set up by writing \\t after the given word representing the function. This does not mean that I as translator can add or delete shortcuts. The "\\t" representing that there is a shortcut assigned to this word cannot be changed or omitted, otherwise Pootle would highlight the problem and would not save such translation. Only editable part was the keys forming the shortcut. For example *New Note\\tCtrl+N* means that *New Note* can be created by pressing Ctrl key and letter N at the same time. This setup is very beneficial for translators, as it gives them option to change accelerators while significantly reducing the risk of creating an error.

However, I did now change any of the accelerators, as I did not find sufficient reasons to do so. Many of the them are already established in Windows (Ctrl+C, Ctrl+V, Ctrl+Q...). The rest is specific to Evernote, but Even in source language,

many times there is no linguistic or other kind of connection of the keys used and the function they put into action (e.g. Ctrl+Alt+Enter for *Presentation mode*). Any changes would yield very little to no benefit and would only increase potential for confusion.

Hotkeys, which are the second group mentioned in 2.3.4, were different case. They are represented by symbol &. First letter after this symbol will serve as hotkey. One phrase or word can have both hotkey and accelerator assigned to it: e.g. *New &Note\\tCtrl+N.* The *New Note* is part of the menu, so the accelerator Ctrl+N will perform the function without the need of going to the menu, and the hotkey *N will* serve for navigation in the menu.

In section 2.3.4 there was emphasis on importance of not creating conflict within keyboard shortcuts. In case of accelerators, the conflict is catastrophic, as one combination would start two function at the same time. In case of hotkeys it is not such a problem. First, since hotkeys serve for navigation, there would have to be two items with same hotkey in the same part of menu. This means that there can be same hotkey used for one item in *File,* one in *Edit,* one in *View*, and so on. Second, even when there is conflict and two items have same hotkey, pressing the hotkey will just switch between the two, which is not at all fatal.

My solution for hotkeys was to create alphabetical list of Czech letters excluding the special characters (as they are not generally used for hotkeys). When I assigned hotkey to particular item, I added the item on the list next to the given letter. This helped me to keep track and come up with solutions for avoiding conflict. However, I soon run out of alphabet, so I started assigning several function to the same letter. That is okay as long as two same hotkeys are not in the same menu.

3.3.5 Choosing Appropriate Language and Style (2.3.5)

Generally, Evernote uses language common in this kind of application. All the functions for text editing have similar or same names as commonly used text editors. Structure of menus and terminology used for each item is similar to the one used in Windows and its commonly used applications. Most of the confusion about meaning was not the issue of poorly chosen style, but was created by lack of context, as argued in 3.3.1.

However, there were few cases where the phrase was prone to more than one

interpretation.

I already mentioned an example of expression *Email notes to* in section 3.3.2. Not only my translation did not fit, but I was also confused about the meaning. *Email notes to* was followed by email address which is assigned to every new user based on his/her username. My first interpretation was “email my notes to following email”. After the investigation of the function, it became clear it is not the case. I was unable assess the function, so I asked the localization manager about the it. He explained that if someone would sent e-mail to the mentioned address, this email would appear as note in the users Evernote account. So if any user needs to save something as a note to his account but cannot access Evernote at the moment, he can just send it to his account via this email. Obviously, this email needs to be Evernote specific, as assigning personal email for this function would cause every single email to be downloaded to every Evernote device that would allow synchronization. The shortest phrase conveying the meaning of this function I could come up with was “Convert emails sent here to notes:”. We can now see that the probable cause of the vague source language expression was also the space restriction. As was stated in 3.3.2, this issue was not resolved, as adding more space will take significant amount of time.

3.3.6 Variables and Composite Strings

It was argued in section 2.3.6 to use composite strings with variables as little as possible, and when the use could not be avoided, Microsoft strongly suggested not to compound several variables. While this sounds great on paper, it seems the reality of my case study was often different. We will now explore the problem using several examples containing more then one variable.

First example contains two variables in on sentence:

Unable to copy the note "<NoteTitle />" to your Evernote account because it exceeds the maximum note size limit of <MaxNoteSize />.

In this case the two variables do not compose such a difficult problem, because the variables in the tags (< >) are self-explanatory. The first pair of tags will “fill in” whatever is the title of the note that is attempted to copy. Since the title is in quotes, the message is clear and there will be no need for adjustment of the word form in Czech based on case. The second pair of tags represents the maximum size of note that is allowed for the given account level. In reality, the message

could look something like this: *Unablte to copy the note "Tasks for next week" to your Evernote account because it exceeds the maxium note size limit of 5 MB;* and the submited translation is: *Nebylo možné kopírovat poznámku „<NoteTitle />" na Váš účet Evernote, protože překračuje maximální velikost poznámky <MaxNoteSize />*.

Second example has three variables and is little more complicated:

Evernote {0} is now available (you have {1}). Would you like to {2} it now?

The issue is more difficult mainly because unlike the first example, the variables offer no explanation whatsoever. I had to guess what they could stand for based on context and I could not be sure about their nature until I encountered the message in the testing phase on running software. It is a message box that appears when there is new version of Evernote (of course the user has to either enable checking for automatic updates or check for updates manually to see this message). First variable stands for the number of the new version (e.g. Evernote 5.7). The second variable is the version that is currently installed by the user (e.g. Evernote 5.6.). The third variable has only two options. It either offers to *download* the new version, or *install* the version, depending on whether it was already downloaded or not. Once the problem is seen in context, it is not so hard to resolve. Example of this message can be seen on Figure 10.

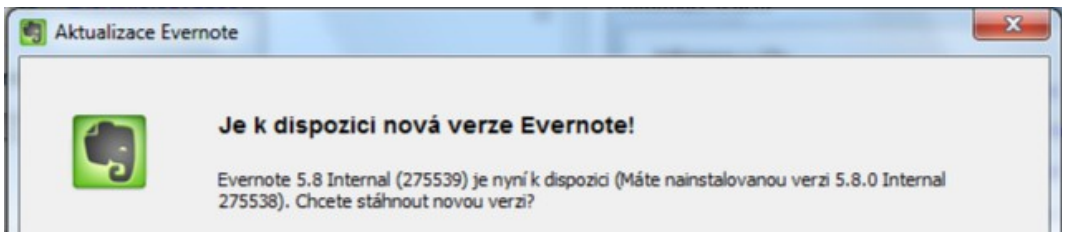

Figure 10 shows composite string of three variables in running application.

Third example has four variables and very little text for explanation:

{0} out of {1} ({2}%)\\n{3} days left in cycle

Since I could not come up with any possible theory, I just translated it very generally and decided to wait for testing phase. It took considerable amount of time before I encountered it. This string can bee seen on Figure 11 (*105 KB z 60 MB*...). It informs user of Basic level of account how much upload is left out of the 60 MB/Month allowance. By the time I was in the testing phase, I already had

Premium account with unlimited upload. I found this after I realized I need to create Basic account exactly for such content. Once I learned what the variables stand for, it was possible to come up with translation, but since there is only one variation of this sentence ending with the word "days", the phrase is grammatically wrong (both in English and Czech) when there is only 1 *day* left in the cycle for English (also 1-4 days in Czech). This supplies the example of situation from section 2.3.8, when translator could be blamed for wrong form of word, which is forced upon him by insufficient internationalization.

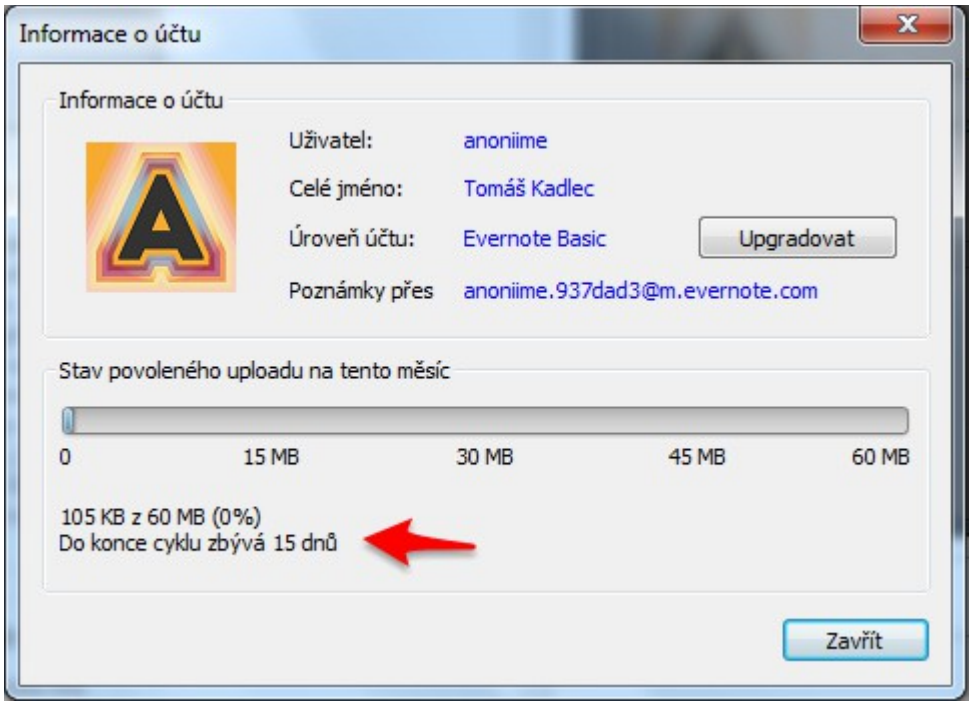

Figure 11 shows composite string of four variables.

This brings us to the last issue I would like to point out from section 2.3.6. It was pointed out that while English has only one singular and one plural form of words, it does not have to be the case for other languages. Czech is good example of such language. In the previous example there was no variation for the singular form of the word *day*. While most of the time there are plural and singular variations available in Pootle as seen on the example bellow, it is still not sufficient for Czech.

Viewing {0} note

Viewing {0} notes

These two variations allow me to translate the first one as *Zobrazuji (jednu) poznámku,* and the second one as *Zobrazuji (pět a více) poznámek*. But there is no variation for the cases of *dvě, tři* and *čtyři poznámky.*

While this is an issue to consider, it is not the kind of problem that breaks the product or causes translation incomprehensible to the users. On the other hand, it supports the statements from section 2.4.2, that localization and internationalization can severely limit translation. We can clearly see here that localization forces translation to operate on level of sentence equivalency – if there is not an equivalent sentence in source language for *2-4 Notes* that is needed in Czech, there is nothing the translator can do to solve it.

3.3.7 Enabling Operating System Support (2.3.7)

Evernote uses the Microsoft Locale Module, so there were no issues with displaying Czech special characters or format of date. Sort orders also work according to Czech rules even when accented characters are included (Á).

There was also nearly full support of special Czech characters. Only issue was no support of Czech quotation marks. This meant that when I pressed the combination for quotation marks, only the English version appeared. After consulting with localization manager, Czech quotation marks were manually added to the Pootle. Typing quotation marks on keyboard still produces the English version, but the button with Czech version is added under that translation window (Figure 12).

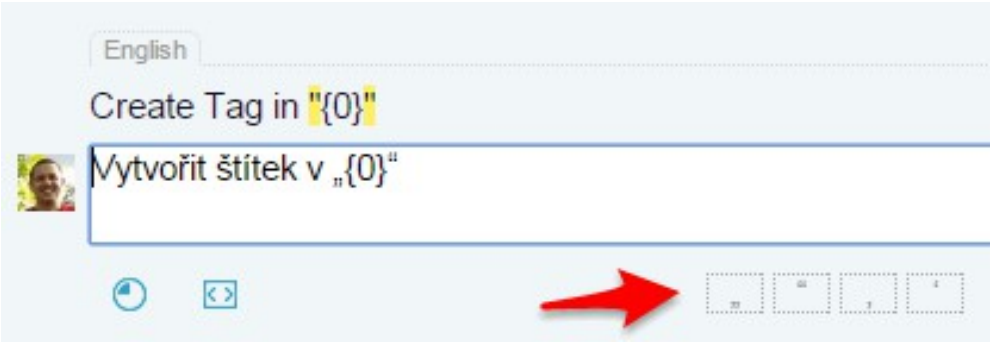

Figure 12 – added buttons for Czech quotation marks.

3.3.8 Translation

Translation phase corresponded with the phase described in section 2.4.2. I have received the source text: it was not sent to me in a file, and I also had an access to it before I could actually translate, as any user can view the strings and *suggest* translations, but after the communication with localization manager of Evernote I

received rights to *submit* translation in Pootle (3.2.1). This meant that I could also review suggestions of other users and either approve or discard them.

When translating, I logged into Pootle and opened Evernote application to potentially search for context. When I could not understand some issues like variables, composite strings, special symbols, mechanics of the keyboard shortcuts, etc., I emailed my questions to the localization manager. Our relationship highly corresponded to the one described in 2.4.4. Direct contact was an advantage which was beneficial both for me and the manager, who had more direct control over the localization. However, I was definitely consuming his time, especially by the end of the project, and since we could communicate directly on chat, sometimes we got off the track of the original problem.

When I started the project, there was roughly 6800 words left for translation. However, the "finished" part of the project was translated by L10N Robot and volunteer users, which meant I had to review and most of the time re-translate all of the 15138 total words. At first glance it seemed that the Terminology database, translated parts, TM, and suggestions from other users would be of great help, but most of the time it was quite the opposite.

After some evaluation, it became clear that available users suggestions needed linguistic knowledge. Most of them followed capital letters of the original (*New Note* was often *Nová Poznámka)* and the translation was usually not following any kind of common terminology of the given Operating System or similar applications. Also, the suggestions where mostly carried over by the L10N Robot from other projects like Android client, where the terminology often differs. Sometimes the available translations from other users did more harm then good. For example for *Print* there was suggestion *Vytisknout*. It is not wrong and it seems adequate, so I accidentally submitted that translation, even though I was using *Tisk* as translation for *Print*. I noticed this inconsistency in the testing phase: in main *File* menu there was *Tisk*, but in *File* menu of a specific note there was *Vytisknout*.

The terminology base consisted of translations of similar quality, all submitted by L10N Robot from other projects or users who translated up to few dozen words, so the level of consistency was minimal. Most of the time I did not use the terminology, as I would have to edit it, which would add to my already large project. Since I was translating Windows client, I used Microsoft Language

Database as the primary source of reference.

After some time I realized that I would have to write long emails to the localization manager about everything I did not find in the software, so I decided to wait for the testing phase. This made sense especially as I was informed about the process – once the Czech build was compiled by developers, any change to the translation would be automatically downloaded as an update, so it would be no inconvenience to anyone and I would not have to wait for someone to actually compile new build, which would be especially ineffective due to the 9 hour time difference.

When the translation phase was close to end, it was becoming more and more clear that I cannot see several functions in context because I have only Basic account. Before I could ask for the Premium for limited time, I received a code with Premium access lasting one month. This function is automated and by then I have reached the specific score needed for the upgraded account.

When I completed the first version of the translation, Evernote team was working on some major update for the software and everyone was really busy. I could not get any answers to issues I did not understand. However, the team of developers still managed to compile (to insert the translated text to the application and set up a process that would continue doing that automatically after changes to translation) Czech build for me. This was time saving move from the localization manager, as I eliminated majority of my questions via testing the compiled build. Any changes I did in the translation implemented automatically and were available in few hours in form of automatic updates.

3.3.9 Evernote localization process in bigger picture

I described my my experience with translation, and now we will look at broader picture. We will review approach to translation, localization and crowdsourcing of the Evernote company. All of the information comes from a Skype conversation with localization manager Pavel Sapronov.

In house or outsourced translation?

I described in the section (2.4.4) that the first question for the localization process is if the translation should be carried by in-house team or whether it should be outsourced. In the case of outsourcing, the decision needs to be made between employing agency or freelance translators. Interestingly enough, Evernote

successively went through all the options. When the app was created and internationalized, the first language it was localized into was Russian, mainly because the owner and founder of Evernote company is from Russia. This localization was carried out by in-house team of employees. After that, Evernote experienced very fast growth on the market, resulting in demand for more, and most importantly fast localization. Short deadlines is the area where the agencies excel because they possess necessary resources in terms of technology and manpower. When this phase was over, disadvantages of agencies emerged: it was not possible to get one point-person for the project. With changing managers and translators came inconsistencies in the localization. Since there was no further need for the fast localization, Evernote localization team started searching for dedicated freelancers who would take up the project and carry out localization of new updates. As the freelancers oversee only one language, they are able to provide consistent localization. As of 2015, in-house localization team consists of only five employees, who oversee these contractors. They are successful in this task and have dedicated freelancer for every of approximately 20 supported languages.

Will the Investment be Worth it?

In section 2.2.2 I also mentioned the term return of investment and presented a survey from 2006 which concluded that only 26% of researched companies had implemented some measures to assess it.

This is definitely not the case of Evernote, as it has clear data on localization and translation return of investment every supported language, and these data are reevaluated regularly. Needles to say, this data is confidential.

Crowdsourcing

Since my translation was outsourced and it was done voluntarily, it falls into category of crowdsourcing (2.4.5). We will now look at Evernote's approach to this issue.

I found article on Evernote blog from 2009 that introduces the whole initiative. By then the application available in Russian and English, but the company decided to localize into other languages. The reactions in discussion under the article represent the general discussion about crowdsourcing we outlined in section 2.4.5. Most people were enthusiastic about the possibility of having their favorite app in their native language, and they were willing to help for free. Few individuals were

worried about the quality, mentioning Facebook as a bad example of community crowdsourcing that yielded poor quality. One person even expressed negative opinion concerning the creation of economical imbalance, and was not happy that Evernote would be selling product that people helped to create for them for free.

My first impression when seeing Pootle and reading this article was that localization of Evernote is mainly crowdsourced project, but in my conversation with the localization manager I learned that quite the opposite is true. All the fully localized projects that are finished and continually supported were translated by paid professionals, except for Portuguese (and now Czech).

Crowdsourcing was not the main focus from the start: the application was internationalized so the professionals could translate it, but once all the mechanisms were in place, it took almost no extra effort to open the translation server to public and let people crowdsource suggestions of translation. However, Evernote is going to focus on crowdsourcing more and it will try to motivate people to contribute more. Some of the mechanisms for motivation are already in place. After certain number of translated characters, the volunteer receives Premium account for a month. Also, anyone can see their score (translation, editing and reviewing suggestions from other users have different point evaluation) and post it on LinkedIn or Twitter. It also rates users by their contribution (Figure 13). However, these mechanism are not promoted, which will change in the future, as Evernote will try to motivate users of non-localized languages to join in.

Figure 13

In section 2.4.5 I also quoted two main concerns about crowdsourcing: quality and commercial business using volunteers for profit.

Evernote localization team is well aware of lower quality and does not expect the crowdsourced localization to equal translation carried out by professionals. The goal is to achieve level of localization that will be sufficient for users to be able to experience most functions of Evernote in their native language.

The second issue is making profit on volunteers. While Evernote is a commercial business, significant proportion of the Evernote's functions are free. One should not forget that Premium level of account provides functions like Optical Character Recognition, which can be obtained elsewhere. Also, one of the major features of Premium is unlimited upload, which can mean that user's notes are occupying significant disk space, which is not connected to the cost of translation. Last point I would argue is that even with fully crowdsourced project, Evernote's employees will still need to invest their work time into coordinating, assessing and finishing the final build. Of course, any of these points do not deny the fact that eventually Evernote could attract more Premium users for whom the language barrier presented significant issue and thus increase its profit. I only wanted to point out that it is not in any case straightforward example of business making profit on free

work of someone else.

Continuous Localization

Last topic that is connected to translation is continuous localization (2.4.6). Evernote is clearly the case of software that is updated fairly often, and most of the updates are not very large. As we already mentioned, Evernote currently supports approximately 20 languages. Localization team has dedicated contractor for each language. If there is any update, the contractors are contacted beforehand so the update in other languages could be as quick as possible.

Pootle is fully automated to separate any new text from the source code within new content and after translation is carried out, the text is implemented back into the code.

3.3.10 Testing

In section on testing (2.4.3) I made some distinctions between linguistic, visual and functional testing. In this case, the words of Keiran Dunne (2009, 212) – "functional testing also often serves as another round of linguistic and cosmetic testing" – were not only confirmed, but taken to the next level, as I was performing all stages of testing almost simultaneously.

I installed newest English version of Evernote on my laptop and then installed the localized internal version on my desktop, so I could compare the functionality in real time and was not forced to switch languages every time I was not sure about the translation. This was possible because one of the featured functions of Evernote is keeping user's notes synchronized across all devices, so any user can log-in on several devices simultaneously. This was helpful especially since I only had one Premium account, which I could use on both computers at the same time.

I also created new account with my secondary email, which helped me with two things. First was to go through the process of account creation and see various message boxes and instructions. For that I tried to create account with my existing email to create errors. After I exhausted these options I created this second account so I could view the content that varies for different account levels. For example, the window with *Account information* for Premium user just lists Premium features, including unlimited uploads, while users of Evernote Basic see how much of the monthly upload allowance they already used and there is option

to upgrade (Figure 11).

It also “helped” that the first version of localization was little unstable, so it crashed several times and I could go through some synchronization errors and message boxes guiding me to fix the problems.

Most of the time I was simply going through the menus as methodically as possible. I found out that some mistakes are really hard to solve once made. Right in the start, in the *File* part of the menu, I noticed there is *Nová Zápisník* (New Notebook), which was supposed to be *Nový zápisník*. Two mistakes in single phrase: wrong form of the word *new* and capital *Z*. Normally I would just find it through full-text search in the Pootle and corrected it, but this time the special symbol “&” for hotkeys was in the middle of the word. This sign naturally does not show in the actual app, and as I just began the testing, I did not realize that I can identify not only existence, but also placement of this symbol, because it is a hotkey. Therefore I had no way of knowing on what position in the phrase *New Notebook* it is placed. Full-text search for word Nová and Zápisník had close to 100 hits each, so it did not narrow the spectrum enough. In the end I finally realized is probably going to be the & symbol and the best idea I came up with at the moment was writing *Nová Zápisník* into the search and placing the & at every position possible. I found the problem after nine searches. After this experience I was able to identify the symbol for hotkeys and found mistakes in full text search much faster.

Another big issue was repeatability. There is option to *search* in notes in the main menu, and then same option in note's window. When I decided to change my translation in the main window I also found the other instance and changed it accordingly, but I was risking making a mistake if there was some other instance in other menu that would require different form of the word.

Full text search in Pootle proved to be the most valuable tool, as the mistakes or typos were usually one of a kind. After some experience I was able to identify special symbols and their probable placement.

After going through the menus and functions under “working” circumstances, I tried to create some errors. I switched of my internet connection and tried to synchronize or access parts that require internet connection, like Market or Announcements. This caused several error announcement to pop up.

Unfortunately there was lack of any resources that would help me trigger all the

errors or see all the message boxes, dialog windows and such. I asked the localization manager for some solutions, but the answer was that they use automatic process only for testing the code errors, but not for localization. However, these are usually in full sentences that provide enough context to be understood even when read in Pootle.

Also, there was no automatic mechanism for testing for conflicting hotkeys. Conflict arises when there are two functions assigned to one letter in the same part of menu. I tried to check all the menus for conflict, but such approach is prone to human error. However, conflicting hotkeys will not create any fatal errors. Pressing hotkey assigned for two menu items will just switch between the two items.

Lastly, I tried to check for some general mistakes which did not cause alert in Pootle. I used the full text search for double space and inserted space in front of question mark, full stop, come, etc.

3.3.11 Unresolved issues

During the testing phase I found out that not everything I translated was inserted in the Czech build. The main client was fully localized, but functions *Presentation* and *Annotation*, which open separate window, are in English. I notified localization manager about the issue, and he found out that Evernote has switched to some new protocol that causes this issue. Even though this is now issue for all the languages, it was not resolved in the nine days since my original notification. Because of that, the Czech localization cannot enter the open beta for user reviews, which – if positive – would allow it to be officially released.

4 Conclusion

The first part of the thesis proved to be sufficient preparation for the case study. Reviewed literature was very relevant and there was not a single issue I encountered in the case study that was not described in the publicatinos. Even fifteen years old *Guide to Localization* by Esselink is still very relevant source, and its main concepts and principles are very helpful when preparing for localization of software. The sources not only described the process of localization and its context, but also provided information about latest trends in the industry.

The case study was very valuable experience for me. At first, I was really thrilled to be working on actual localization, but I have to say that after working on the localization for significant amount of time, I strongly identify with many observations of Anthony Pym: translation in localization process is very limited and straightforward, there is no room for creativity, and the target language often cannot be expressed in all its grammatical rules. Localization of software is fascinating process, but translation itself is stripped to the basic core of equality on level of words and sentences.

The main goal of the localization being officially released was not achieved. Unexpected issues in the area of software development have occurred, delaying the public beta release by at least a week. As of now, the developers plan to resolve this issue, and release the localized version to public for review. If the reviews will be satisfactory, the Czech version will be officially released. The most optimistic estimation is that the beta could be released on Evernote's forums by May 12, 2015.

5 List of References

Common Sense Advisory. 2009. *Freelance Translators Clash with LinkedIN over Crowdsourced Translation.* Accessed April 28, 2015. http://www.commonsenseadvisory.com/Default.aspx?Contenttype=ArticleDetAD&tabID=63&Aid=591&moduleId=391

Common Sense Advisory. 2015. *Survey of 3,000 Online Shoppers Across 10 Countries Finds that 60% Rarely or Never Buy from English-only Websites*. Accessed April 22, 2015. https://www.commonsenseadvisory.com/Default.aspx?Contenttype=ArticleDet&tabID=64&moduleId=392&Aid=21500&PR=PR

Daily Crowdsource. 2015. *What is Crowdsourcing?*. Accessed on April 10, 2015 http://dailycrowdsource.com/training/crowdsourcing/what-is-crowdsourcing

DePalma, Donald A. 2006. "Quantifying the return on localization investment." In *Pespectives on Localization,* edited by Keiran J. Dunne, 15–37. Amsterdam/Philadelphia: John Benjamins Publishing Company.

Dunne, Keiran J. 2009. "Assesing software localization: For a valid approach." In *Testing and Assessment in Translation and Interpreting Studies*, edited by Claudia V. Angelelli and Holly E. Jacobson, 185–225. Amsterdam/Philadelphia: John Benjamins Publishing Company.

Esselink, Bert. 2000. *A practical Guide to Localization.* Amsterdam/Philadelphia: John Benjamins Publishing Company.

European Comission. 2012. *Studies on Translation and Multilingualism: Crowdsourcing Translation.* Luxembourg: Publications Office of the European Union. PDF. Accessed April 15, 2015 at: http://bookshop.europa.eu/en/crowdsourcing-translation-pbHC3112733

Evernote. 2009. *We want to speak your language.* Accessed April 29, 2015. https://blog.evernote.com/blog/2009/07/29/evernote-translation-program/

Evernote. 2010. *Evernote Trademarks.* Accessed May 1, 2015. https://evernote.com/trademark/

Evernote 2015. *Getting Started with Evernote for Windows Desktop.* Accessed May 1, 2015. https://evernote.com/evernote/guide/windows/#1

Getlocalization. 2015. *Continuous Localization*. Accessed May 2, 2015. https://www.getlocalization.com/continuous-localization/

Golota, Hanna. 2015. *To Outsource or not to Outsource Localization: Pros and Cons.* Accessed May 1, 2015. http://www.globalme.net/blog/to-outsource-or-not-to-outsource-localization-weighing-the-pros-and-cons

Intel Developer Zone. 2013. *Go Global at the Speed of Innovation with Continuous Localization.* Accessed May 2, 2015. https://software.intel.com/en-us/blogs/2013/07/01/go-global-at-the-speed-of-innovation-with-continuous-localization

Lingo Systems. 2004. *The Guide to Translation and Localization, Preparing for the Global Marketplace.* Idaho: MultiLingual Computing.

LinkedIn. 2015. Group *Translators again Crowdsourcing by Commercial Businesses.* Accessed April 19, 2015. https://www.linkedin.com/groups/Translators-against-Crowdsourcing-Commercial-Businesses-2032092?home=&gid=2032092&trk=anet_ug_hm

LISA. 2003. *Localization Industry Primer, 2nd edition.* Accessed April 19, 2015. http://www.dynamiclanguage.com/home14/wp-content/uploads/2014/07/LISA-Globalization-Primer.pdf

Microsoft Language Portal. 2015. *Search Terminology.* Avaliable at https://www.microsoft.com/Language/en-US/Default.aspx

Microsoft. 2015. "Globalization Step-by-Step." Accessed April 20, 2015. https://msdn.microsoft.com/en-us/goglobal/bb688113

Nielsen, Karin. 2015. *Localization Outsourcing versus Inhouse.* Accessed May 1, 2015. http://www.codexglobal.net/general/localization-outsourcing-versus-inhouse/

Pym, Anthony. 2001. *Localization and Linguistics.* SLE Conference Avaliable at: http://usuaris.tinet.cat/apym/on-line/translation/translation.html

Pym, Anthony. 2004. *The Moving Text: Localization, translation, and distribution.* Amsterdam/Philadelphia: John Benjamins Publishing Company.

Tomala, Anna M. 2013. "Localization Vendor Management." In *Localisation Management: Core Focus.* Accessed May 1, 2015. http://www.multilingual.com/downloads/coreFocus139.pdf

Zajiček, Štefan. 2013. *Lokalizačný proces pri vývoji software.* Bachelor Thesis, Masaryk University.

6 Anotace

Autor: Tomáš Kadlec

Katedra: Katedra anglistiky a amerikanistiky, FF UPOL

Název česky: Případová studie lokalizace softwaru: Evernote

Název anglicky: Software Localization Case Study: Evernote

Vedoucí práce: PhDr. Pavel Král

Počet znaků celkem: 90 248

Anotace

Cílem této práce bylo shrnout poznatky odborné literaury o procesu lokalizace softwaru a představení tohoto termínu v celém jeho kontextu, zaměřit se na roli a pozici překladatele v tomto procesu, a následně provést případovou studii lokalizace. V rámci případové studie byla provedena lokalizace softwaru Evernote.

Práce potvrdila, že k tématu existuje široká škála odborné literatury, která se vyjadřuje ke všem klíčovým prvkům procesu lokalizace a poskytuje východiska k řešení běžných problémů tohoto procesu. Cílem případové studie bylo vytvořit lokalizaci softwaru Evernote, která bude použita jako oficiální česká lokalizace této aplikace. Tohoto cíle nebylo dosaženo z důvodů nečekaných komplikací ve zdrojovém kódu aplikace, které způsobily zpoždění celého procesu. Je možné, že po vyřešení těchto problému bude daného cíle dosaženo.

Klíčová slova: globalizace, internacionalizace, lokalizace softwaru, překlad, crowdsourcing,

Abstract

The goal of this thesis was to review process of software localization, introduce this term in its broader context, focus on role and position of translators in the process, and finally carry out case study of software localization. For the purposes of the case study, the author localized software called Evernote. The purpose of the case study was to create localization of the software that would be used as official Czech localization. This goal was not reached because of certain unforeseen complications within the source code of the application. It is possible that after when these issues are solved, this goal may be reached.

Keywords: globalization, internationalization, localization of software, translation, crowdsourcing,