Localisation Focus

THE INTERNATIONAL JOURNAL OF LOCALISATION

ISSN 1649-2358

Centre for Next Generation Localisation

Fondúireacht Eolaíochta Éireann
Science Foundation Ireland

The peer-reviewed and indexed localisation journal

VOL. 11 Issue 1

PUBLISHER INFORMATION

Editor: Reinhard Schäler, *Director*, Localisation Research Centre, University of Limerick, Limerick, Ireland
Production Editor: Karl Kelly, *Manager* Localisation Research Centre, University of Limerick, Limerick, Ireland
Published by: Localisation Research Centre, CSIS Department, University of Limerick, Limerick, Ireland

AIMS AND SCOPE

Localisation Focus – The International Journal of Localisation provides a forum for localisation professionals and researchers to discuss and present their localisation-related work, covering all aspects of this multi-disciplinary field, including software engineering, tools and technology development, cultural aspects, translation studies, project management, workflow and process automation, education and training, and details of new developments in the localisation industry. Proposed contributions are peer-reviewed thereby ensuring a high standard of published material. Localisation Focus is distributed worldwide to libraries and localisation professionals, including engineers, managers, trainers, linguists, researchers and students. Indexed on a number of databases, this journal affords contributors increased recognition for their work. Localisation-related papers, articles, reviews, perspectives, insights and correspondence are all welcome.

To access previous issues online go to http://www.localisation.ie/resources/locfocus/pdf.htm and click on the issue you wish to download. Use the following logon details - username: locfocsub and password: LFVxii01

Subscription: To subscribe to Localisation Focus - The International Journal of Localisation www.localisation.ie/lf

Localisation Focus – The International Journal of Localisation (ISSN 1649-2358) is published and distributed annually and has been published since 1996 by the Localisation Research Centre, University of Limerick, Limerick, Ireland. Articles are peer reviewed and indexed by major scientific research services, including: Bowker, Cabell's Directories and St Jerome Publishing Translation Studies Abstracts Online. It is also included in the Library of Congress Collections.

FROM THE EDITOR

What's New?

Localisers are always looking for the latest advancements and innovative approaches to localisation. Their ideal world would be the one were the project management triangle, also known as the 'iron' triangle, of cost, time, and quality could be squared into an approach where the ideal product or service could be produced in no time for free – in any of the world's languages. Of course, this is not possible. However, some of the research presented in this volume, will go a long way to support localisers looking for solutions in the core areas of localisation complexity, among them encoding, consistency, post-editing, accessibility, and standards.

Murhaf Hossari and Arthur Cater, two researchers from University College Dublin (UCD), present their work on Pattern-based Enhancements to the Unicode Bidirectional Algorithm which, as it stands, can be problematic when displaying bidirectional script.

Joss Morkens from Dublin City University (DCU) reports on a Mixed-method Study of Consistency in Translation Memories in an attempt to verify the assumption that the use of Translation Memories (TMs) promotes consistency in translation.

An international research group consisting of Silvia Rodríguez Vázquez (TIM/ISSCO Geneva) and Jesús Torres del Rey (Universidad de Salamanca) discusses a Communicative Approach to Evaluate Web Accessibility Localisation Using a Controlled Language Checker. While their theoretical framework has not yet been tested empirically, their approach of using a language-based accessibility validator has already yielded promising results.

Every localiser has struggled with the use of shortcut keys in different locales, an issue addressed by the paper presented here by three researchers from Vilnius University's Institute of Mathematics and Informatics: Gintautas Grigas, Tatjana Jevsikova, and Agné Strelkauskyté.

In the new age of Machine Translation (MT), post-editing is one of the central issues for localisers. It is addressed by Celia Rico of the Universidad Europea de Madrid who proposes a Flexible Tool for Implementing Post-Editing Guidelines.

A collaboration between Microsoft and the Localisation Research Centre at the University of Limerick produced the last paper in this issue, written by Asanka Wasala (LRC/UL), Dag Schmidtke (Microsoft), and Reinhard Schäler (LRC/UL) who report on a detailed analysis of two of the most widely known localisation industry standards, the open XLIFF standard and the Microsoft proprietary LCX standard.

The results presented by the international research teams in this issue of Localisation Focus will not square iron triangles. However, they demonstrate that localisation research is alive and well – in an environment where academia-industry collaboration is not the exception but the rule.

Please support us in distributing this research as widely as possible and encourage your colleagues to submit their localisation research to the next issue of Localisation Focus.

Reinhard Schäler

Pattern-based Enhancements to Unicode Bidirectional Algorithm

Murhaf Hossari, Arthur W S Cater
UCD School of Computer Science and Informatics,
UCD Dublin,
Belfield,
Dublin 4, Ireland
murhaf.hossari@gmail.com, arthur.cater@ucd.ie

Abstract

In this paper, we present an improvement upon the Unicode Bidirectional Algorithm, eliminating the need for manually added directional codes in many cases. The modified algorithm recognizes cases conforming to four general patterns, and provides the correct directionality to their constituent characters without the need to use directional codes. Experiments performed on 593 paragraphs used in Apple software localised for Arabic showed that this approach succeeds in 86.3% of our defined cases correctly (the recognized four general patterns), which constitutes 54.3% of the total paragraphs in our test set. We also present other wrongly displayed cases requiring future work.

Keywords: *Unicode Bidirectional Algorithm, Right-to-Left Text Layout, Internationalisation, and Localisation*

1. Introduction

Internationalisation and Localisation are the processes that make it possible for users around the world to use software in their own languages. Often, different character sets (different scripts) are needed to write text in different languages, and some languages are read and written right-to-left rather than left-to-right. Unicode is now widely used in software when there is a need to deal with multiple scripts. Unicode provides standard unique codes for characters belonging to different scripts. It is also responsible for character representation in computer software. The Unicode Bidirectional Algorithm is responsible for representing scripts that have different horizontal directions (Gross 2006, Khaddam and Vanderdonckt 2011, Unicode 2012b).

Most scripts have left to right directionality. This means that the order by which the characters are stored in memory, also called logical order, corresponds to the order of by which the characters are displayed in a left-to-right sequence. This left-to-right order is called the visual order. However, for some scripts such as those used to write Arabic, Hebrew, Farsi, Urdu and others, the characters are written and read from right to left. The logical order of characters in memory is the reverse of their visual order. Users cannot be expected to accept software that needs them to type strings backwards (Atkin and Stansifer 2004, Unicode 2012c).

The issue gets more complicated when text contains scripts that have different directionality properties e.g. Arabic text with embedded English fragments, or vice versa. This is very common in localised software and this is exactly when the bidirectionality issue arises.

The Unicode Bidirectional Algorithm ("Bidi Algorithm") takes logical order strings as its input and reorders the characters to produce the visual order. It provides this functionality based partly on the nature of individual characters - whether they are right to left or left to right characters - and partly on clear specifications and rules dealing with digits, punctuation and other symbols. Frequently it succeeds in producing the correct visual order (IBM Corporation 2006, Abdelhadi et al 2011).

For example, the sentence "Doubt is a pain too lonely to know that faith is his brother" can be displayed according to the following different layouts:

- Left-to-right layout: *Doubt is a pain too lonely to know that faith is his brother.*
- Right-to-left layout: *.rehtorb sih si htiaf taht wonk ot ylenol oot niap a si tbuoD*
- Bidirectional layout: *Doubt is a pain too lonely to know rehtorb sih si htiaf taht.*

However, the Bidi Algorithm does sometimes fail to display the bidirectional text as it was intended by the user. Unicode provides various directional formatting codes to overcome these cases, where the ordering of characters can be manually adjusted in order to override the default behaviour of the Bidi Algorithm and thus show the text in the correct way (Unicode 2012c).

In this paper we describe our approach to improve the behaviour of the Bidi algorithm, by first finding and classifying the situations where the Bidi algorithm needs the addition of directional formatting codes to show the text correctly, and then providing solutions that can help the algorithm produce the correct character order without the need for directional codes.

We first analyze a large number of cases that previously needed directional formatting codes in a data set taken from Apple software localised for Arabic. Next, we classify these cases into different groups based on their format. Finally, we modify the Bidi algorithm to handle cases in each group so that the correct bidirectional display layout is shown. This modification does not change the overall behaviour of the algorithm in the vast majority of the correctly displayed cases.

The rest of this paper is organized as follows: in Section 2 we review the Unicode Bidirectional Algorithm and the problematic cases that it faces. In Section 3 we explain four patterns of these problematic cases which we handled in our approach. In Section 4 we list the evaluation methods we followed. Section 5 presents possible use of our modifications to the Unicode Bidirectional Algorithm. Section 6 presents our future work and Section 7 provides the conclusions.

2. Unicode Bidirectional Algorithm

The Bidi Algorithm reorders characters in a bidirectional text in order to produce the correct visual representation of the text. Text has a direction that is the general flow - letters and words in sentences - where English language for example has left-to-right flow, Arabic and Hebrew have right-to-left flow. Mixed text contains left-to-right characters and right-to-left characters and it flows in both right and left directions according to characters' type when it is displayed. That is why reordering of characters is needed (Unicode 2012c).

To achieve the reordering, The Bidi Algorithm defines a set of rules to extract the directionality property of each existing character and to deal with the logical order of characters in order to reach the visual order. It proceeds in multiple phases as will be briefly explained:

In the first phase it separates the text into multiple paragraphs. It then runs the following phases on each paragraph. Splitting the paragraphs is done by detecting a paragraph separator at the end of the paragraph. In the second phase, each character of the paragraph is annotated with its directional type. Directional types can be categorized in three main groups:

- Strong types, which include left-to-right characters (e.g. Roman characters), right-to-left characters (e.g. Arabic characters).
- Weak types: mainly numbers, number separators and unit symbols.
- Neutral types: most punctuation marks and white space character.

In the third phase, the directional types are transformed into the corresponding embedding levels by following an ordered sequence of rules. Each rule resolves a certain directional type (weak, neutral, white space,...etc) into either a right-to-left or left-to-right embedding level. At the end of this phase, all weak and neutral types will have either right or left direction. In the fourth phase, the characters are reordered according to the resolved embedding levels from the previous stage. It can be then displayed correctly (Ishida 2003, Unicode 2012c). An example to illustrate the different stages of Unicode Bidirectional Algorithm is given below.

The term Embedding level denotes a simple way to refer to the directionality of the characters where even and odd levels refer to left-to-right and right-to-left directions respectively. It also indicates the nesting depth of the text, the level increases when text nesting goes deeper. Deeper levels are explicitly added by using directional codes (override and embedding format codes). Levels start from 0, which is left-to-right. Maximum level is 61.

The term Paragraph embedding level (Base level) denotes the paragraph direction, usually determined by the first strong type character of the paragraph but possibly explicitly set. The code R indicates a strong right-to-left character; L indicates a strong left-to-right character; N indicates a neutral character, and

WS indicates a white space. Spaces between directional types and embedding levels are just for clarification

For expository purposes, upper case is used to refer to right-to-left characters in the input and output text of the example which now follows.

Logical (memory storage) Order: *CANNOT CONNECT TO SERVER* "*mail server name*"

Paragraph segmentation will result in recognizing the text as one paragraph. Base level is set to right-to-left because the first character is right-to-left.

Firstly, the Bidi algorithm transforms the text into directional types:

RRRRRR WS RRRRRRR WS RR WS RRRRRR WS N LLLL WS LLLLLL WS LLLL N

Secondly, the algorithm uses the following rules to resolve embedding levels:

- Neutrals between two left-to-right types get left-to-right direction, and similarly for right-to-left.
- Neutrals between left-to-right and right-to-left types get the embedding direction, which is usually the direction of the paragraph (first strong character direction) unless it is forced otherwise.

RRRRRR R RRRRRRR R RR R RRRRRR R R LLLL L LLLLLL L LLLL R

Thirdly, the algorithm sets the embedding levels to the corresponding values:

111111 1 1111111 1 11 1 111111 1 1 2222 2 222222 2 2222 1

Finally, the algorithm reorders the embedding levels accordingly and displays the correct visual order:

"mail server name" REVRES OT TCENNOC TONNAC

2.1 Problematic Cases

The Unicode Bidirectional Algorithm displays the correct layout for a text in most of the cases. There are various reasons why it sometimes does not, such as (1) assuming that the paragraph direction is the direction of the first strong character, (2) complicated nesting of strings of different types that cause neutral/weak characters to be misinterpreted, (3) Strings with a special nature such as part numbers. Rarely, the string is ambiguous even for human eye (Ishida 2003, Unicode 2012c).

To solve these problematic cases, explicit directional codes are manually added to the problematic text to enforce the correct layout. Those codes provide the help that Unicode Bidirectional Algorithm needs by defining a separate embedding level, adding an invisible strong-type character, or overriding directionality (Ishida 2003, W3C 2007, Unicode 2012c).

Directional codes defined as a global standard by Unicode are:

- U+202B RIGHT-TO-LEFT EMBEDDING (RLE)
- U+202A LEFT-TO-RIGHT EMBEDDING (LRE)
- U+202C POP DIRECTIONAL FORMATTING (PDF)
- U+200F RIGHT-TO-LEFT MARK (RLM)
- U+200E LEFT-TO-RIGHT MARK (LRM)

While using the directional codes can enforce the intended layout, there are some pitfalls for using them:

- They are manually added, and the fact that they are invisible can lead to mistakes.
- They are not trivial to use. They demand a good understanding of how the algorithm works which is not the case most of time.
- When localising software, translators will not always know how strings will be shown at runtime, strings can be dynamically composed from many substrings. Sometimes the only way to know there is a problem is to wait until it is reported. Then the translator fixes it with directional codes and then tries again.
- Being invisible, the Unicode characters in text are always in the danger of getting removed or changed whenever the text is modified or transferred among different environments.

In HTML, a few other ways were added to fix the problematic cases by creating new HTML tags that deal with bidirectional text. A property can be added to the HTML tag, that contains text, that will will be able to force the direction to either right-to-left or left-to-right rather than using the Bidi Algorithm

implicit direction detection way. New embedding levels can be created to solve certain cases by adding special directionality HTML tags. Some new work is being done for HTML5 and new tags are being created for better support of bidirectional text (Lanin 2011). However, in our approach we propose a solution which is not related to a specific platform, moreover, it eliminates the need to add directional codes and HTML tags.

شركة أمريكية متعددة الجنسيات Apple

Figure 1. Layout without directional codes (incorrect layout)

3. Patterns of Problematic Cases

Underlying the cases in which the Unicode Bidirectional Algorithm fails to produce the correct visual order, there are phenomena which are more common than others. We studied a large number of cases of bidirectional text, which are not displayed correctly in order to spot the most frequent cases. We used Apple software localised for Arabic, which has a significant amount of bidirectional strings and paragraphs. We first extracted the problematic cases by collecting the strings and paragraphs that contain directional codes. We assume that these strings have display problems as these directional codes would have been added manually by localisers to fix problems. Adding directional codes does not necessarily mean that the text is shown incorrectly by the Bidi Algorithm, due to the fact that strings might contain variables that are replaced with strings at runtime, so localisers sometimes add directional codes in order to guarantee that the layout would be correct regardless of what the variable was replaced with (Ishida 2003).

Studying these strings gave us closer look at the problematic cases that the Bidi Algorithm fails on. According to these cases we extracted the following frequently occurring causes:

1 Incorrect Paragraph Direction: this is the most common problem. It happens when a paragraph gets an incorrect direction (embedding level) due to its first character. As a result, the base level is set to a wrong direction. For example an Arabic paragraph that has the first word in English should have a right-to-left direction but because the first character is English, the paragraph is assigned a left-to-right direction by the Bidi algorithm.

Example:
A string belongs to this class will show incorrectly as in figure 1.

The correct layout is shown in figure 2.

For simplicity, will use lowercase for English and uppercase for Arabic Logical order:

apple IS AN AMERICAN MULTINATIONAL CORPORATION

Visual order by Unicode Bidirectional Algorithm:

apple NOITAROPROC LANOITANITLUM NACIREMA NA SI

The correct visual order:

NOITAROPROC LANOITANITLUM NACIREMA NA SI apple

Apple شركة أمريكية متعددة الجنسيات

Figure 2. Layout with directional codes (correct layout)

This is usually fixed by adding a directional code “RLM (Right to left mark)” at the beginning of the paragraph. Because RLM has a right-to-left strong type, the paragraph will get a right-to-left direction and the problem is solved. However, other higher-level protocols are sometimes applied to the Unicode Bidirectional Algorithm where the paragraph direction is determined by the maximum number of characters of each directional type (Unicode 2012c).

We apply a mix of rules in order to improve the way that the Unicode Bidirectional Algorithm determines the paragraph direction. These rules consist of our defined rules which depend on last character and word count to determine paragraph direction combined with the rules that are already defined by the Bidi algorithm (first character and character count). The rules work as follows:

- If the majority of words and the majority of characters have the same directional type then:
    - The paragraph direction is the same as the direction of the majority of words.
- If the majority of words has different direction than the majority of characters then:
    - If the first and last strong character have the same direction then the paragraph direction is set to their direction.
    - If the first and last strong character have different directions then the paragraph direction is the direction of the majority of words.

However, these rules do not solve all the cases but we find that it improves the Algorithm's sense of the language of the paragraph. A possible alternative solution is to use more complicated language detection techniques but that is not suitable for use in the Bidi Algorithm as performance is vital when rendering text. Another possible solution, which is specific to localised software, deduces the language of the paragraph based on the language package containing it. For example, if the paragraph is taken from the Arabic localised content package, then treat it as a right-to-left paragraph.

2 Embedded contra-flowing bracketed text: this second cause is that the paragraph contains embedded text with the opposite direction which is surrounded by quotation marks, parentheses, brackets, or the like, and this embedded text also contains punctuation and/or more deeply embedded bidirectional text.

Example1:

Logical order: the application is *"SOME ARABIC NAME!"*

Visual order by the Bidi Algorithm: the application is *"EMAN CIBARA EMOS!"*

The correct visual order: the application is *"!EMAN CIBARA EMOS"*

The exclamation mark belongs to the right-to-left string and should show to its left. Translators might fix this either by adding an RLM directional code right after the exclamation mark, or by surrounding the right-to-left string (including the exclamation mark) with a pair of right-to-left embedding codes RLE and PDF.

Example 2:

Logical order: the application is *(NAME, co)*

Visual order by Unicode Bidirectional Algorithm: the application is *(EMAN, co)*

The correct visual order: the application is *(co ,EMAN)*

Similar solution as used in example 1 can be used to solve example 2.

The solution for this issue is based on the observation that material surrounded by parentheses, quotation marks, brackets or the like is usually a single coherent piece of text that is not tightly related to its surroundings. For this, we consider the surrounding characters to have some sort of balancing features where it usually has an opening character and closing one in a proper text. In case this scenario was detected, we treat balanced characters as having stronger combinatory feature than other punctuations within the text they surround. This means that these balanced characters define a separate segment that contains the balanced characters and the text within them. In our approach, we process this segment separately by the original Bidi algorithm. We then combine the result of this segment with the result of the

surrounding text. This would produce the correct layout in most cases. Applying this to the examples above:

Example 1:

Logical order: *the application is "SOME ARABIC NAME!"*

Our algorithm will parse the string first and when the balanced characters are found it will be segmented into two strings:

- *the application is*
- *"SOME ARABIC NAME!"*

Processing these two strings by Unicode Bidirectional Algorithm rules will produce the following visual order:

- *the application is*
- *"!EMAN CIBARA EMOS"* (On its own this is a right-to-left string so the exclamation mark is placed correctly)

Combining those two segments will lead to the intended display of the whole string:

the application is *"!EMAN CIBARA EMOS"*

Example 2:

VPN (PPTP) قطع الاتصال

Figure 3. Correct Visual Order.

Logical order: *the application is (NAME, co)*

Segmentation of the string in a similar way:

- *the application is*
- *(NAME, co)*

Processing each of the string and combining the results yields:

the application is (co ,EMAN)

Unicode is gathering feedback for a proposed enhancement for the bidi Algorithm that will help solving this case. The proposed algorithm is "Bidi Parenthesis Algorithm" (Unicode 2012a). The proposed enhancement will be either applied to the original bidi algorithm or will be added as higher-level protocol that can be used when needed.

3 Embedded contra-flowing partly-bracketed text: The third cause is somewhat similar to the previous in that it also contains balanced characters, but here the problem occurs with the display of the balanced characters themselves. It occurs when the paragraph contains a substring with an opposite direction to the paragraph direction (base direction), the substring is composed of some characters surrounded by balanced characters and some others not surrounded by them and they are either following or preceding them.

Possible directional representations of the case:

RRR LLL (LLLLLL) RRRRR

RRRRR RRR "LLLL" LL RRR

LLL LLL RRR {RRRRR RRR RR} LLL LL

LLLLLLLLL LL [RR] RRRRRRRR LLL LLLLL

Example
For simplicity, we will again use lowercase for English and uppercase for Arabic

Logical order: *CANNOT FIND SERVER pop "mail.me.com"*

Visual order by the Bidi Algorithm: *"pop "mail.me.com REVRES DNIF TONNAC*

The correct visual order: *pop "mail.me.com" REVRES DNIF TONNAC*

Directional codes may be used by translators similarly as before, either by adding an LRM after the last quotation mark or by

surrounding the fragment (pop "mail.me.com") with a left-to-right embedding code pair (LRE and PDF).

The solution we propose is to first diagnose this case by character-level parsing of the paragraph, segmenting it accordingly, performing the standard Bidi Algorithm on each of the segments, and finally combining them to get the expected result.

Applying this to the example:

Logical order: *CANNOT FIND SERVER pop "mail.me.com"*

After parsing the string and detecting that it belongs to this class, segmentation will produce two strings:

- CANNOT FIND SERVER
- *pop "mail.me.com"*

By applying the standard Bidi Algorithm on each of the strings:

- *REVRES DNIF TONNAC*
- *pop "mail.me.com"*

Combining the results will produce*: pop "mail.me.com" REVRES DNIF TONNAC*

4 URLs ending in '/' embedded in a right-to-left paragraph: This fourth cause occurs when website URLs containing '/' as a last character are embedded in a right-to-left paragraph and not followed by a left-to-right character.

Example:

Logical order: *CANNOT FIND WEBSITE http://www.me.com/mail/*

Visual order produced by the Bidi Algorithm: */http://www.me.com/mail ETISBEW DNIF TONNAC}*

The correct visual order: *http://www.me.com/mail/ ETISBEW DNIF TONNAC*

Translators usually fix this by adding an LRM directional code after the URL in order to enforce a left-to-right direction on the '/'.

The solution we propose is to first detect URLs that meet the conditions of this class, then to segment the paragraph in a way that the URL creates a separate segment, then perform the Bidi Algorithm on each of the segments. Back to the previous example:

Logical order: *CANNOT FIND WEBSITE http://www.me.com/mail/*

After parsing the string and detecting that it belongs to this case, segmentation will produce two strings:

- CANNOT FIND WEBSITE
- http://www.me.com/mail/

By applying the standard Unicode Bidirectional Algorithm on each of the strings:

- ETISBEW DNIF TONNAC
- http://www.me.com/mail/

Combining the strings back together will produce: *http://www.me.com/mail/ ETISBEW DNIF TONNAC*

4. Evaluation

We applied the proposed solutions on strings used in Apple software localised for Arabic. Extracting the text that contains Unicode directional codes helped us in analyzing the cases where the Unicode Bidirectional Algorithm needs help to provide the correct layout. However, the data we used is property of Apple and is not currently available for public use.

We extracted 593 paragraphs that need Unicode directional codes to produce the correct visual order. Many of these cases contain variables that are replaced with real values at runtime due to the fact that the data is used in localised software. Variables can stand for usernames, computer names, server names, number of pictures, minutes, hours, etc.

Having those variables in the paragraphs as they are when evaluating may lead to ambiguity or misinterpretation of the results. Variables usually consist of one or few neutral characters which means that applying the Unicode Bidirectional Algorithm to the paragraph with the variables might hide some

cases that cause problems for the layout at runtime, when they are replaced by strong type characters. In addition, some variables are surrounded by Unicode directional codes even though, while looking at the variable itself, it is not apparent to us why it needs directional codes. For these reasons, we replaced the variables with representative data and tried to cover the possibilities for the values that a variable might be replaced with.

Possible replacements included randomly created words in left-to-right characters, right-to-left characters and numbers with neutrals. Also the replacements can have various sizes according to specified ranges. As we are dealing with Arabic localised content, it is most probable that adding left-to-right characters would create more problems.

Case	Number of Occurrences
Case 1	254
Case 2	57
Case 3	49
Case 4	13
Total	373

Table 1. Number of paragraphs in each recognized case in the evaluation set.

Afterwards, we classified the paragraphs, each class representing one of the types of case that we are attempting to fix. Recognizing each case and grouping them permits counting how frequent each type of case is; it also helped us detect the cases that did not fall under any of the types we defined and which would require future work. Of the 593 paragraphs, we found 254, 57, 49 and 13 paragraphs of the first, second, third and fourth types respectively: a total of 373, or 62.9%. Table 1 shows the number of paragraphs in each recognized case.

We used the Java reference code provided by Unicode (Unicode 2009) for implementing the original Bidi Algorithm. It processes one paragraph at one run. We ran our tests on the original Unicode Bidi Algorithm using strings containing directional codes that translators had added, and on the modified algorithm using strings with those directional codes removed. If for some string the modified algorithm provides the same embedding levels and ordering in its output, without the help of directional codes, as the original algorithm provides with that help (discarding the directional codes when comparing), that is a success for the modified algorithm.

The results we found were that 254 paragraphs fell under the first case, 57 paragraphs fell under the second case, 49 paragraphs fell under the third case and 13 paragraphs fell under the forth case. Which made a sum of 373 recognized cases. The tests revealed that the modified algorithm succeeded in 322 cases, that is, 86.3% of the 373 recognised as instances of general classes, and 54.3% of the 593 strings.

5. Possible Use of the Modifications to the Unicode Bidirectional Algorithm

The Bidi Algorithm is an established global standard that is widely used for the purpose of displaying bidirectional text. For this reason, changing the standard is a difficult decision to be made. An alternative is to apply the modified version we propose in order to detect the cases of the types listed and add the correct directional codes to the text before processing by the standard Bidi Algorithm. In this way adding directional codes can be done automatically for these cases and reduce the danger of errors caused by manually adding them.

6. Future Work

During our analysis of the data that needs directional codes to provide the correct visual order of characters, we encountered four other productive patterns that possible solutions could readily be provided for. Those cases include:

1 File names written in right-to-left characters: When naming a file using right-to-left characters it will cause problems in layout because file extensions are using latin characters with a period which can be misplaced in layout. Also when displaying file paths problems exist because paths might contain left-to-right folder names or disk names (Microsoft MSDN 2012).

Example

Figure 4. Visual order by Unicode Bidirectional Algorithm

Logical order: *NAME.doc*

Visual order by Unicode Bidirectional Algorithm: *.docEMAN*

Correct visual order: doc.EMAN

2 Unit symbols and mathematical signs in right-to-left text: Numbers in right-to-left scripts have a left-to-right flow, however, mathematical signs and other neutral unit symbols should be displayed in a certain layout in those scripts and this usually causes problems.

Example:

Logical order for the string (*-2%*) in a right-to-left script: *-2%*

The visual order by Unicode Bidirectional Algorithm is the same: *-2%*

The correct layout should be: *%2-*

3 Phone numbers in right-to-left scripts: As previously mentioned, numbers in right-to-left scripts have a left-to-right flow, however, the overall layout of the script would be right-to-left which creates problems when formatting phone numbers (or any similar concept of numbers such as part numbers,...etc)

Example:

Logical order: *PHONE NUMBER IS +987 (65) 432 1098*

Visual order by Unicode Bidirectional Algorithm: *1098 432 (65) 987+ SI REBMUN ENOHP*

Correct visual order: *+987 (65) 432 1098 SI REBMUN ENOHP*

4 A list of bidirectional items separated by a neutral: When there is a sequence of bidirectional named items (items named with a mix of left-to-right and right-to-left characters) separated by punctuation or any neutral character, the Bidi Algorithm tends to place same-direction characters together in an embedding level, punctuation and all.

Example:

If there was a list of application names separated by commas where some applications have Arabic names and others have English names.

Logical Order: *CONTACTS, TOOL airport, finder, PREVIEW, safari*

Visual order produced by Bidi Algorithm: *safari ,WEIVERP ,airport, finderLOOT ,STCATNOC*

Correct visual order: *safari ,WEIVERP ,finder ,airport LOOT ,STCATNOC*

This problem is harder to detect, because establishing that a bidirectional text makes a list of items separated by a punctuation mark cannot be done simply. It needs more complicated tests such as parsing the words based on dictionaries or large corpus to get an idea about the text and whether it looks like separate items/names separated by punctuation marks or it is just a regular cohesive text with punctuation marks. However, if separators were a set of punctuation marks it would be easier to detect those lists, because normal text should not usually contain two commas or two dashes adjacent to each other.

Finally, adding these improvements to the Unicode Bidirectional Algorithm will introduce a new challenge. Algorithms that are responsible for text processing and rendering have to be very quick. Performance is a vital part when thinking about those algorithms. Improvements to the algorithm by applying sentence and word segmentation will add more rules and processing, which in turn will decrease the efficiency of the algorithm. We used regular expressions to detect certain patterns. Optimizing the way the regular expressions are used can improve the performance of our improved version of the algorithm. Working on improving the efficiency of the patterns detection techniques is an important avenue for future work.

7. Conclusions

When displaying bidirectional text that contains characters in both left-to-right and right-to-left scripts, the Unicode Bidirectional Algorithm is used

as a global standard and a widely used tool to achieve this purpose. It reorders the characters based on the order they were typed in (Logical Order) using several rule-based phases of processing. It displays the characters after they have been reordered in the correct readable way (Visual Order) most of the time. Sometimes, however, it needs the help of invisible directional codes to explicitly force the direction of a character or a string of characters. Analysis of the cases where the Unicode Bidirectional Algorithm fails to show the correct visual layout allowed identification of a few patterns that fit many of these problem cases. Solutions are provided for four of those patterns, and a modified algorithm is found to show the correct visual layout in the majority of those cases. Using as evaluation data 593 strings extracted from Apple Arabic localised software, experiments showed that the modified algorithm corrected the layout of 86.3% percent of the cases that were fitted by patterns, 54.3% of all the cases that frustrated the original Unicode Bidirectional Algorithm.

We also presented problematic patterns that we did not address in our current work. Future work will aim to extend our approach to handle these cases, to find still more patterns and solutions in order to improve the overall behaviour of the Unicode Bidirectional Algorithm.

References

adMob (2010) AdMob Mobile Metrics [online], available: http://metrics.admob.com/ [accessed 15 Aug 2010]

Abdelhadi, A., Mouss, L. H. and Kadri, O. (2011) 'Efficient Algorithms for the Integration of Arabic Language in Mobile Phone', International Journal of Computer and Electrical Engineering Vol.3, No.3, June 2001, pp. 379-383.

Atkin, S. and Stansifer, R. (2004) 'Implementations of Bidirectional Reordering Algorithms', 18th International Unicode Conference, Hong Kong.

Gross, S. (2006) 'Internationalization and Localization of Software', Master Thesis, Eastern Michigan University.

IBM Corporation (2006) 'National Support Guide and Reference', AIX 5L Version 5.3 , Publication No. SC23-4902-03.

Ishida, R. (2003), 'What you need to know about the bidi algorithm and inline markup', ITSC, available: http://www.w3.org/International/articles/inline-bidi-markup/bidi.pdf

Khaddam, I. and Vanderdonckt, J. (2011) 'Flippable User Interfaces for Internationalization', EICS'11, Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems, New York, pp 223-228.

Lanin, A. (2011) 'Additional Requirements for Bidi in HTML', W3C, available: http://www.w3.org/International/docs/html-bidi-requirements/

Microsoft MSDN (2012) 'Formatting Control Characters', available: http://www.microsoft.com/middleeast/msdn/control.aspx\#Samples

Seng J. (2001) 'Internationalisation and Localisation of the Internet', Synthesis (Singapore), available: http://www.i-dns.net/pdf/internationalisation_localisation.pdf

Unicode (2009) 'Reference code implementing Unicode Bidirectional Algorithm' , available: http://www.unicode.org/Public/PROGRAMS/BidiReferenceJava/

Unicode (2012a) 'Bidi Parenthesis Algorithm', available : http://www.unicode.org/review/pri231/

Unicode (2012b) 'The Unicode Standard', available: http://www.unicode.org/versions/Unicode6.0.0/

Unicode (2012c) 'Unicode Bidirectional Algorithm', Unicode Standard Annex #9, Unicode 6.0.0, available : http://www.unicode.org/reports/tr9/

W3C (2007) 'Unicode control vs. markup for bidi Support', available: http://www.w3.org/International/questions/qa-bidi-controls.en.php

A Mixed-methods Study of Consistency in Translation Memories

Dr. Joss Moorkens
Centre for Next Generation Localisation
School of Applied Language and Intercultural Studies, Dublin City University
www.cngl.ie
Joss.Moorkens2@mail.dcu.ie

Abstract

Translation Memory (TM) has become widely-used since the early 1990s, its use based on several assumptions: that it saves time, provides cost savings, and maximises consistency. The purpose of this research is to develop a method for measuring consistency in TMs, and to use this method to interrogate English-to-German and English-to-Japanese TMs from the localisation industry in order to find out whether the use of TM tools does, in fact, promote consistency in translation.

The research uses an explanatory mixed-methods approach. In the quantitative phase, translation units are categorised based on whether the TM-based translation process had introduced consistency or inconsistency. The research found inconsistencies of letter case, spacing, and punctuation in source texts, and inconsistent terminology, formatting, and punctuation in target texts. In a follow-on qualitative phase, thirteen interviews were conducted with translators and others with experience of TMs who confirmed that the findings from the quantitative phase corresponded with their experiences.

Keywords: *Translation memory, localisation, consistency, quality, fuzzy matches, terminology, mixed methods*

1. Introduction

Language service providers (LSPs) are under increasing pressure to provide large quantities of localised content with a fast turnaround and at low cost. This has led to increased use of technology, particularly translation memory tools, throughout the localisation industry. The axioms behind the use of TM tools are that they reduce the cost of translation, save time, and minimise inconsistency. They allow the opportunity to leverage legacy translations and have been shown to increase the productivity of translators, thus saving time and cost (Somers 2003, p42). Costs are further reduced as translators are often paid based on TM match analyses, with full payment offered for translation from scratch, partial payment offered for editing fuzzy matches, and a small (or sometimes no) fee paid for reviewing 100% matches. In theory TM tools should aid the production of consistent translations as previously translated work is recycled. This research aims to discover whether this is true in practice.

While there has been some research on the introduction of errors (Rieche 2004; García 2008) and error propagation in TMs (Bowker 2005; Ribas López 2007), there appears to be little research on consistency in TMs. This research proposes to find a method of interrogating TMs for consistency, then to apply that method to measure and categorise inconsistencies in sample TMs in a case study.

This research uses a sequential explanatory mixed methods design, beginning with a quantitative study of TMs in the first phase, and following with a series of qualitative interviews in the second phase. The interviews are intended to explain and add richness to the quantitative results, demonstrating whether the findings from the first phase are applicable to TMs generally in the experience of translators and other TM users from the localisation industry. This paper will focus largely on the quantitative phase of the study. The intended outcomes of this research are to find a method of measuring inconsistency in TMs, to show what types of inconsistencies (if any) commonly occur in TMs, and to suggest methods of minimising inconsistency in translation using TM tools.

2. Methodology

We use a sequential, explanatory mixed methods design. Creswell and Plano Clark state that this design is appropriate to "when a researcher needs

qualitative data to explain significant results" (2007, p72). The follow-up explanations variant of this design is used in this study, in which the qualitative data are intended to expand upon the quantitative results. In the first, quantitative phase of the study, four sets of TM data collected from two large and world-renowned technology companies were measured for consistency. The data includes two English-to-German TMs and two English-to-Japanese TMs as detailed in section 2.2. In the second phase, qualitative interviews with translators and other translation professionals who work with TMs were conducted to explore their experiences of consistency issues in TM.

2.1 Quantitative phase

The quantitative phase of the study measures segment-level and sub-segment level inconsistency. Segment level inconsistencies are observed where two segments that one could reasonably expect to be formally identical differ from each other in some way. We view source segments as being formally different if their meanings differ, but use the term 'inconsistent source segments' to refer to cases where there are very minor formal differences between two source segments and such differences do not reflect any semantic differences between the segments in question. Such minor formal differences include differences in capitalisation, tags, punctuation, spaces, character formatting, and spelling (where a segment may be inconsistent with another segment simply because of a misspelling, inconsistent use of British or US English spelling, or a typographical error in one of the segments).

In the case of target segments, it appears reasonable to expect segments that are translations of 'the same' source segment (i.e. segments that are translations of different tokens of the same source type) to be formally identical, especially in a translation memory scenario where the goal is to reuse existing translations for already encountered source segments.

Where there are two different translations (and thus two different target segments) for a single source segment type, this is considered a target segment-level inconsistency. As there may be more than one inconsistency within these segments, each discrete inconsistency is counted and categorised. The differences between the target segments in question can be very minor formal differences (as defined above), but they can also be more substantial, in extreme cases even leading to semantic differences between the two segments.

	Source Text	Target Text
Category 1	Callouts window	Fenster "Callouts"
inconsistent source-> inconsistent target	Callouts Window	Callouts-Fenster
Category 2	Plane, perspective	Ebenen, perspektivische
inconsistent source-> consistent target	Planes, perspective	Ebenen, perspektivische
Category 3	Camera button	Kameraknopf
consistent source-> inconsistent target	Camera button	Kamera- Schaltfläche
Category 4	text background	Texthintergrund
consistent source-> consistent target	text background	Texthintergrund

Table 1. Example of TU Categories

At segment level, the following four categories are possible:

1. inconsistent source segments are translated as inconsistent target segments
2. inconsistent source segments are translated as consistent target segments
3. consistent source segments are translated as inconsistent target segments
4. consistent source segments are translated as

consistent target segments

The current study is primarily interested in categories 1 and 3, but also in the possibility of consistency being introduced during the process of computer-assisted translation (category 2). Category 4 may be seen as the ideal in specialised translation, whereby the TM has provided the best possible leverage and thus saved the maximum possible amount of time and money. An example of each category from our TM data is given in Table 1.

Inconsistent segments are counted by identifying the number of types n. The number of segment-level inconsistencies is the type count minus one (n-1). Thus in the case of a single source segment (type) that has 4 tokens, if there are 3 separate translations (3 types; one of which appears twice), then the number of target segment inconsistencies is 2 (or 3-1). We give a special status (of 'master' or 'reference' segment) to one of the target segments, and treat the other two segments as inconsistent with that reference segment. The reference segment is the one which appears first in our sorted list, and which a translator could have, but did not reuse in unchanged form. For example, the following four translations for 'Click an empty part of the drawing area.' appear in the TM data:

a. Klicken Sie auf der freien Zeichenfläche.
b. Klicken Sie auf einen freien Bereich der Zeichenfläche.
c. Klicken Sie auf einen freien Bereich der Zeichenfläche.
d. Klicken Sie auf einen beliebigen freien Bereich auf der Zeichenfläche.

Although there are four tokens, there are only three types: a, b, and d. If we assign the status of reference segment to segment a, the segments that are inconsistent with the reference segment are b (repeated for c) and d: thus we count two inconsistencies. When we have three types n=3, and since our count is of type (n - 1), we count two inconsistencies.

At segment-level, source or target segments are either consistent or formally differ and are thus inconsistent. However, there may be more than one inconsistency within these segments. For this reason we also count and categorise sub-segment inconsistencies found within inconsistent target text segments (those found in categories 1 and 3 above).

These inconsistencies are categorised mostly per part of speech aside from those with inconsistent punctuation or where word order has been changed. If there are more than three inconsistencies within a target segment, we consider that segment to have been wholly retranslated. These categorised inconsistencies may be further subcategorised; for example nominal inconsistencies that differ lexically, or in number (singular/plural). Other typical sub-segment inconsistency categories are verb, space, punctuation, preposition (for German), and postposition (for Japanese).

These subsegment-level inconsistencies are counted in the same way as segment-level inconsistencies: we identify the number of types n, assign one the status of master or reference segment, then count the types that are inconsistent with the part-of-speech or word order in the reference segment. Thus the count is n minus the reference segment (n-1). Again, the reference segment is the one which appears first chronologically, and which a translator could have reused, but chose to edit in some way.

2.2 Research data

The data used in the first phase of this study is four sets of TM corpora obtained from two companies who gave permission for their data to be used. All four TMs were presented in the TMX format, parsed using a Python script, copied into the LibreOffice Calc spreadsheet tool and categorised as per section 2.1.

TM A is an English-to-German TM containing 22,691 TUs of aligned segments of which 188 contain only numbers, dates, or punctuation symbols. The remaining 22,503 TUs were categorised as per the typology in section 2.1. TM B is an English-to-Japanese TM from the same project as TM A, containing 18,799 TUs. After removing those segments that contain only numbers, dates, or punctuation symbols, 18,650 TUs remained to be categorised. TM C is an English-to-German TM containing 301,583 TUs. After removing those that contain only numbers, dates, or punctuation symbols, 293,924 TUs remained. TM D is an English-to-Japanese TM containing 298,700 TUs from the same project as TM C. After removing the TUs that contain only numbers, dates, or punctuation symbols, 292,258 TUs were left.

For TMs C and D, a sample of 50,000 TUs was studied. In order to confirm homogeneity between the sample and the full TM corpus in each case, a chi square test was carried out using Excel. The test was

based on comparative measurements of corpus statistics as measured using Wordsmith Wordlist software and comparative measurement of the frequency of category 3 inconsistencies. The corpus statistics used were types (distinct words), standardised type-token ratio, and mean word length (in characters). The chi square test found no significant difference between the sample and the full TM.

2.3 Qualitative phase

The second phase of this research is a series of qualitative interviews with translators and others in the localisation industry with experience of using TM tools. Calls for potential interviewees were circulated via email and Twitter, and translators were approached at several industry events. Details of interviewees may be seen in Table 2. These were in the form of face-to-face personal interviews or, where this was not possible, telephone interviews, seeking opinions on results and conclusions reached in the quantitative phase of the study. The interviews were recorded, transcribed to a LibreOffice document, and coded using NVivo qualitative analysis software. As is typical when coding with Nvivo, emergent themes were gathered as free or open codes (or nodes, to use the Nvivo terminology). These open codes were then sorted into a hierarchy of branching "tree nodes" to reflect the "structure of the data" (Bazeley 2007, p100).

3 Quantitative results by category

	Job Title	First Language
Interviewee A	Translator	Brazilian Portuguese
Interviewee B	Translator	German
Interviewee C	Translator	Spanish
Interviewee D	Translator	French
Interviewee E	Translator	Brazilian Portuguese
Interviewee F	QA Specialist	English
Interviewee G	QA Specialist	Spanish
Interviewee H	Language Technology Consultant	English
Interviewee I	Project Manager	German
Interviewee J	Project Manager	Japanese
Interviewee K	Chief Operating Officer	Spanish
Interviewee L	Workflow Manager	French
Interviewee M	Software Localisation Engineer	Spanish

Table 2. Interviewees from Qualitative Phase

3.1 Category 1 TUs (Inconsistent ST segments with inconsistent TT segments)

3.1.1 Category 1: Inconsistent source text segments

Table 3 shows the most commonly-occurring types of category 1 ST inconsistency (actual number of inconsistencies in parentheses).

Category 1 TUs contain minor inconsistencies (as specified in our typology in section 2.1) in the source segment and other kinds of inconsistencies in the aligned target segment. The number of category 1 TUs found in our four sets of TM data differed, but in all four TMs the most prevalent category of source text (ST) inconsistency was in letter case or capitalisation of words. None of the TT segments aligned with ST segments that contain inconsistencies in letter case themselves contain instances of inconsistent letter case; rather the TT segments in question evince other kinds of inconsistencies, as in example 1 (with differences highlighted in bold) from TM C:

	TM A	TM B	TM C	TM D
Letter case	34% (60)	37% (13)	66% (314)	72% (753)
Punctuation	28% (48)	8% (3)	13% (60)	7% (73)
Tags	24% (42)	29% (10)	10% (46)	13% (137)
Typo	2% (4)	3% (1)	4% (17)	1% (11)
Space	11% (20)	23% (8)	8% (37)	7% (68)
Word Order	0	0	0% (1)	0
Total Segments	370	65	995	1980
Total Subsegment Inconsistencies	174	35	475	1042

Table 3. Inconsistencies found in Category 1 Source Text Segments

1.1s (SHIFT+right-click the drawing area.)	1.1t (Klicken Sie bei gedrückter UMSCHALTTASTE mit der rechten Maustaste in den Zeichenbereich.)
1.2s (**Shift**+right-click the drawing area.)	1.2t (Klicken Sie bei gedrückter UMSCHALTTASTE mit der rechten Maustaste in den **Zeichnungsbereich**.)

Example 1

In both TT segments the ST word 'shift' has been translated as 'Umschalttaste' (shift key) and capitalised. This would suggest that a TM match was used despite the change of case in the second ST segment (1.2s). However, the German translation of 'drawing area' was changed from 'Zeichenbereich' to 'Zeichnungsbereich'. According to the metadata, segment 1.1t was created on December 22nd 2006 and last changed two years later on December 7th 2008. Segment 1.2t was created by a different translator on January 15th 2009 and last changed one year later on January 18th 2010.

We found a high number of inconsistent placing of the space character in TM D. These spaces were initially noticed by automatically comparing the ST segment and the following, seemingly identical, ST segment, as 54 of the 68 space inconsistencies were at the end of the segment following a full stop. Again, the aligned target text (TT) segments contain other kinds of inconsistencies, as in the following example:

The ST segment contains a space inconsistency,

2.1s {1}lsp{2} file.	2.1t {1}lsp{2} ファイルから自動的にロードされます。
2.2s {1}lsp{2} file. [Contains extra space]	2.2t {1}lsp{2} ファイルは変更しないでください。

Example 2

while the aligned TT segments differ by particle (は 'wa' and から 'kara'), 2.1t has the additional adjective automatic or 自動的 'jidouteki', and the verbs differ semantically and in form. These TT segments also provide examples of explicitation in Japanese TT, a phenomenon that appears throughout TMs B and D. The translation of an ST noun has become a sentence in the Japanese TT, containing detail not present in the ST. Thus we have 'It will be automatically loaded from the lsp file.' in segment 2.1t and 'Please do not change the lsp file.' in segment 2.2t.

3.1.2 Category 1: Inconsistent TT segments

In Table 4, the number of subsegment inconsistencies may be seen to be larger than that in Table 3. This is because a segment may contain up to three subsegment inconsistencies before we consider it completely retranslated. Among category 1 TUs we found a large proportion of noun inconsistencies, comprising between 35% and 48% of the total number. For example, in TM C there were 323 noun inconsistencies of which 12 showed influence of the source language in one instance but not in another, and 87 contained inconsistencies of capitalisation or case, as per the following example:

Example 3 also displays a phenomenon that accounts for the high prevalence of preposition inconsistencies in TM C. We found 138 preposition inconsistencies in category 1, just over 19% of the total. 126 of these preposition inconsistencies (and thus 18% of the total) are secondary changes as required by the change of noun, thus we see an alternation between the phrases 'in der Befehlszeile' (in the command line) and 'an der Eingabeaufforderung' (at the command prompt).

	TM A	TM B	TM C	TM D
Noun	35% (84)	45% (15)	44% (323)	48% (616)
Punctuation	1% (3)	15% (5)	8% (57)	10% (129)
Tags	2% (5)	21% (7)	0% (2)	11% (147)
Verb	19% (45)	9% (3)	9% (66)	6% (75)
Space	3% (6)	0	5% (38)	11% (141)
Word Order	17% (41)	3% (1)	11% (81)	0% (3)
Preposition	5% (12)	N/A	19% (138)	N/A
Particle	N/A	3% (1)	N/A	3% (42)
Completely rewritten (Not added to total)	12	5	9	12
Total Segments	370	65	995	1980
Total Subsegment Inconsistencies	240	33	730	1291

Table 4: Inconsistencies Found in Category 1 Target Text Segments

3.2 Category 2 TUs (Inconsistent ST segments with consistent TT segments)

Category 2 TUs contain ST inconsistency and thus introduce consistency in the TT. The majority of these ST inconsistencies in all TMs analysed were inconsistent letter case. Example 4 from TM C is typical of this ST inconsistency.

3.1s At the Command prompt, enter subtract.	3.1t Geben Sie in der Befehlszeile differenz ein.
3.2s At the **c**ommand prompt, enter subtract.	3.2t Geben Sie **an der Eingabeaufforderung DIFFERENZ** ein.

Example 3

Although capitalisation of the first letter of a German language noun means introduced consistency would be expected in example 4, there are instances of the same TM, containing a ST space inconsistency similar to those found in all four sets of TM data, we see a German noun in lower case:

	TM A	TM B	TM C	TM D
Letter case	44% (140)	49% (219)	46% (480)	45% (505)
Space	30% (95)	21% (96)	28% (287)	22% (247)
Punctuation	21% (67)	1% (4)	9% (89)	14% (153)
Inconsistent word (Noun)	3% (11) (3 noun)	22% (98) (86 noun)	14% (146) (70 noun)	17% (194) (77 noun)
Typo	1% (2)	0% (1)	2% (19)	2% (21)
Word Order	0% (1)	0% (1)	0% (4)	0
Total Segments	613	914	2077	1801
Total Subsegment Inconsistencies	316	450	1043	1123

Table 5: Inconsistencies Found in Category 2 Segments

Inconsistent punctuation usually has to do with the presence or absence of commas or full stops in the ST which may or may not be retained in the TT. Example 6, from TM D, contains a punctuation inconsistency,

4.1s Action Macro	4t Aktionsmakro
4.2s **a**ction **m**acro	4t Aktionsmakro

Example 4

ST letter case being retained in the TT in all of these TMs (roman lettering is sometimes used in the Japanese TT), particularly if the ST segment is in upper case. This means we have a mix of transposed ST punctuation or formatting and native TT formatting in TT segments. In example 5 from the but also contains an example of a section that has been marked out in the TT, followed by a comment by the translator, explaining that he chose the term 塗

5.1s {1}securityoptions {2}	5t {1}sicherheitsoptionen{2}
5.2s {1}securityoptions{2}	5t {1}sicherheitsoptionen{2}

Example 5

り潰し色 'nuritsubushiiro' for filling in colours. This comment was subsequently propagated within the TM.

There are a number of reasons why ST inconsistencies may be ignored by a translator who chooses instead to accept a fuzzy match. Foremost among these in Japanese is the presence of plurals in the ST. In our study of English to German TMs, plurals did not register in our categories, as we considered that the ST had formally changed and thus accepted that the TT would be inconsistent. However, as there is no distinction between singular and plural in Japanese – numbers are given explicitly or are implicit in context – we can expect to see plural and singular nouns translated consistently in the Japanese TT, and this is indeed the case. Of 76 cases of inconsistent nouns in the ST segments of category two TUs in TM B, 42 differ in number: singular in one case, plural in another, as in example 7.

6.1s {1} If None (Color) is selected as the Map Type then the color needs to be selected.	6t {1} [マップの種類]として[###塗り潰し色]を選択した場合は、色を選択する必要があります。■3-(B037)「なし」という選択肢はなく、「塗り潰し色」という選択肢が表示されるので、このようにしました。(Koizumi 06/11/21)
6.2s {1} If None (Color)) is selected as the Map Type then the color needs to be selected.	6t {1} [マップの種類]として[###塗り潰し色]を選択した場合は、色を選択する必要があります。■3-(B037)「なし」という選択肢はなく、「塗り潰し色」という選択肢が表示されるので、このようにしました。(Koizumi 06/11/21)

Example 6

7.1s Dimension	7t 寸法
7.2s Dimensions	7t 寸法

Example 7

3.3 Category 3 TUs (Consistent ST segments with inconsistent TT segments)

Category 3 contains TUs with inconsistent TT segments, where inconsistency has been introduced in the TM data. Again, the most prevalent category of TT inconsistency was noun inconsistency, as may be seen in Table 6. In TM A we found 81 inconsistently translated nouns (47% of the inconsistencies) of which 18 showed influence of the English source language in one instance as in example 8.

This alternation between 'Border' and 'Rand' occurred three times in the TT and was one of several patterns that emerged within the data.

The Japanese TT in TM B again showed detail being added in translation that was not in the ST. We found ten inconsistencies that were translations of the word 'selecting', as shown in example 9.

In example 9, it is taken as the reference translation as it appeared first in the TM data. Each TT segment contains the noun 選択 (*sentaku*) meaning 'selection' but most involve further explicitation, adding the particle の to make the genitive case. 9.1t is エレメントの選択 or 'selection of elements'. Segment 9.2t is コールアウトエレメントの選択 or 'selection of callout elements'. While this explicitation may make the TT segments clear and understandable, it has a negative effect on leverage. It may be in this case that the first translation contained added detail that was not appropriate for the subsequent translations or that the translators felt that more detail was necessary in the context of the finished document.

After noun inconsistencies, the next most prevalent category in TM B is verb inconsistencies. Of the 40 verb inconsistencies, 18 of them contained another repeated pattern, alternating between using the verb 拘束する (*kousoku suru*) meaning 'to bind or restrict' in one case, and the verb 関連付ける (*kanren*

	TM A	TM B	TM C	TM D
Noun	47% (81)	38% (49)	49% (282)	35% (365)
Verb	20% (34)	31% (40)	5% (30)	6% (59)
Punctuation	7% (12)	6% (8)	8% (44)	18% (183)
Space	1% (1)	1% (1)	11% (61)	27% (272)
Explicitation	1% (1)	5% (6)	1% (3)	3% (32)
Word Order	3% (5)	3% (4)	3% (17)	2% (16)
Preposition	4% (7)	N/A	20% (112)	N/A
Particle	N/A	12% (15)	N/A	6% (57)
Completely rewritten (Not added to total)	3	7	9	35
Total Segments	390	239	826	1713
Total Subsegment Inconsistencies	174	129	570	1035

Table 6: Inconsistencies Found in Category 3 Segments

8s All lines that have been converted using the {1}Create surface borders{2} function can be recognized easily since they are drawn with the {3}Border{4} pen.	8.1t Alle Linien, die mit der Funktion {1}Flächenränder anlegen{2} konvertiert wurden, können Sie leicht erkennen, da sie mit dem Stift mit der Bezeichnung {3}Border{4} gezeichnet werden.
8s All lines that have been converted using the {1}Create surface borders{2} function can be recognized easily since they are drawn with the {3}Border{4} pen.	8.2t Alle Linien, die mit der Funktion {1}Flächenränder anlegen{2} konvertiert wurden, können Sie leicht erkennen, da sie mit dem Stift mit der Bezeichnung {3}**Rand**{4} gezeichnet werden.

Example 8

tsukeru) meaning 'to relate' in another.

In example 10 segment 10.1t translates as 'bind ISO elements to XML elements using object information', segment 10.2t as 'relate ISO elements to XML elements using object information'. Looking through the metadata, each verb choice is not attributable to a single user ID, but the translations using **拘束する** were all saved to the TM at the same time on April 22nd 2009. Two uses of **関連付ける** were also saved then, but all others were dated from the 7th of May in 2009. At that stage, one would presume, the TM tool used must have suggested the previously translated TT segment as a 100% match.

The other Japanese data, TM D, also contain a repeated pattern, alternating between the borrowed English word レイヤ and the native Japanese word 画層 'gasou' 41 times as per example 11.

TM D contains a large number of punctuation inconsistencies. Many of these (23) are marked out

9s Selecting	9.1t エレメントの選択
9s Selecting	9.2t コールアウトエレメントの選択
9s Selecting	9.3t アセンブリの選択
9s Selecting	9.4t 多角形の選択
9s Selecting	9.5t 線の選択
9s Selecting	9.6t 楕円の選択
9s Selecting	9.7t 選択
9s Selecting	9.8t 長方形の選択
9s Selecting	9.9t ベジエ曲線の選択
9s Selecting	9.10t めねじの選択
9s Selecting	9.11t おねじの選択

Example 9

10s Binding ISO Elements to XML Elements using Object Info	10.1t オブジェクト情報を使用して ISO エレメントを XML エレメントに拘束する
10s Binding ISO Elements to XML Elements using Object Info	10.2t オブジェクト情報を使用して ISO エレメントを XML エレメントに関連付ける

Example 10

11s A new layer group filter can be nested only under another group filter.	11.1t 新しいレイヤグループ フィルタは、他のグループ フィルタに対してのみネストできます。
11s A new layer group filter can be nested only under another group filter.	11.2t 新しい画層グループ フィルタは、他のグループ フィルタに対してのみネストできます。

Example 11

12s Accesses Dimensioning mode	12.1t 寸法記入モードにします
12s Accesses Dimensioning mode	12.2t 寸法記入モードにします。

Example 12

13s Annotative objects may have multiple {1}scale representations{2}.	13.1t 異尺度対応オブジェクトには複数の{1}尺度表現{2}がある場合があります。
13s Annotative objects may have multiple {1}scale representations{2}.	13.2t 異尺度対応オブジェクトには、複数の{1}尺度表現{2}を持つものもあります。

Example 13

using the # symbol, others have inconsistently placed quotation marks, and many show indecision as to whether or not to retain ST formatting for commas or full stops as in example 12.

The high rate of preposition inconsistency in TM C is again a secondary effect of noun inconsistency as shown previously in example 3. The inconsistencies of particle in Japanese are also often secondary to a change in verb or verb form from active to passive or, as in example 13, required by verb choice with the 表現 (scale representation) taking the particle 'ga' when the verb 'aru' (to exist) is used, and the direct object particle 'wo' with 'motsu' (to hold).

3.4 Category 4 TUs (Consistent ST segments with consistent TT segments)

These TUs are those that we consider to have been translated consistently. By looking at the number of repeated TUs that fall into this category, we can see the overall rate of introduced TT inconsistency within a TM as per table 7.

4. Qualitative results summarised

Various causes of TM inconsistencies were suggested in the qualitative phase of the study. The influence of clients may have a bearing at several stages of the translation process, such as in deciding on whether to standardise source text, whether to use terminological resources, whether or not to edit 100% matches, and whether to specify in a style guide what format to use in target text. Translator choice, particularly when TMs are shared, or when inexperienced translators are involved, was another suggested cause of inconsistency. Interviewees also felt that the TM tool may not prevent the introduction of inconsistency, citing problems with inappropriate segmentation, tag or formatting issues, insufficient integration of terminology, and a lack of integrated QA checks.

Interviewees felt that the focus of clients is usually on time and cost savings, as a result of which much of the ST that they receive is hurriedly written and ambiguous or inconsistent. This was always seen as negative. All but one of the interviewees said that they had seen many instances of ST inconsistencies as found in categories 1 and 2. They said that educating their customers about the effect of inconsistent ST on the translation process is one way in which they attempt to minimise inconsistency. Interviewee F, a QA specialist, said: “If they (clients) can't control their source text, then we can't be expected to control the target text for them”.

Ambiguous ST segments may be translated in different ways for different contexts and thus 100% matches proposed by the TM tool may not be accurate in each instance. The interviewees would like “maximum consistency” (interviewee K) yet have problems with clients’ assumptions that all 100% matches can be automatically accepted. Eight interviewees said that 100% matches may be erroneous, a point previously made by Reinke (2004). Several interviewees (particularly non-translators) felt that some TT inconsistency may be necessary. F said that, for him, it is more important for a translation to be “accurate and natural and fluent than it is for the resulting translation unit to be recyclable infinitely in all other documents”, adding that this loss of leverage is “a sacrifice we have to make”.

The interviewees felt that explicitation, such as in example 9, may be required depending on the context. Becher has suggested that where there is

	TM A	TM B	TM C	TM D
Category 3 TUs	390	239	826	1713
Category 4 TUs	6674	4263	18343	25541
Total TUs with repeated ST segments	7064	4502	19169	27254
Percentage of TUs with introduced inconsistency	5.5%	5.3%	4.3%	6.3%

Table 7: Introduced Inconsistency in all TMs

explicitation there will always be a reason, not least to "minimise the risk of misunderstanding" (2011, p215/216). The interviewees agreed that there were occasions where this risk of misunderstanding made explicitation, such as that found in the English to Japanese TMs, necessary. They also said that languages with gender often require changes or additions due to "gender agreement issues" (M). Accordingly, explicitation was not always seen as negative.

5. Conclusion

The TMs in this study contain inconsistencies in the source text (category 1), introduced consistency in some of the corresponding target text (category 2), and introduced inconsistency in target text segments (category 3). In the ST, these inconsistencies (such as those of letter case and punctuation) were not found to make any semantic change, and resulted in lost leverage, with an associated cost to the client. A lower match value also leaves an opportunity for the translator to edit the TT. TT inconsistencies may be further propagated, again with an associated impact on time and cost.

Each TM corpus in this study shows a large proportion of introduced noun or term inconsistency in category 3. Many noun inconsistencies demonstrate influence from the source language, and different translation decisions have been propagated throughout the TM, such as the alternation between レイヤ 'laya' and 画層 'gasou' [layer] from TM D (example 11), or between the alternated whole phrases 'in der Befehlszeile' [in the command line] and 'an der Eingabeaufforderung' [at the command prompt] in TM C (example 3). The interviewees in the qualitative phase agreed that these inconsistencies are common in their experience of TM. Term inconsistencies are often propagated as terms from different times or different departments may be contained in one TM. Interviewees also said that an inexperienced translator may easily upload a new version onto the TM "and there's nothing that stops it". F (QA specialist) said that he sees this regularly and suggested that it may be the result of translators working independently without adequate terminological guidance. Interviewees also said that, in Japanese, the current trend for phonetic translation means that TMs often contain inconsistencies between kanji and katakana terms.

In each of the TM corpora there appears to be a lack of clarity as to whether ST punctuation and formatting should be replicated or replaced by that native to the TT, leaving a combination of both in the TM data. This may have been further confused by inconsistent letter case in the ST segments. The interviewees suggested rigid adherence to a style guide as a way of minimising these inconsistencies. While they felt that more could be done within the TM tool environment to minimise inconsistency, they highlighted the importance of ST standardisation, terminology management, TM maintenance and the use of bespoke, targeted TMs as methods of minimising TT inconsistency.

Acknowledgements

This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. Thanks to Dr. Sharon O'Brien in DCU for advice on this paper.

References

Bazeley, P. (2007) *Qualitative Data Analysis with NVivo*. London /Thousand Oaks, CA: Sage Publications.

Becher, V. (2011) *Explicitation and Implicitation in Translation: A Corpus-Based Study of English-German and German-English Translations of Business Texts*, unpublished thesis (PhD), Universität Hamburg.

Bowker, L. (2005) '*Productivity vs quality? A pilot study on the impact of translation memory systems*', Localisation Focus, 4(1), 13-20.

Creswell, J. W. and Plano Clark, V. (2007) *Designing and Conducting Mixed Methods Research*. London /Thousand Oaks, CA: Sage Publications.

García, I. (2008) '*Translating and Revising for Localisation: What Do We Know? What Do We Need to Know?*' Perspectives: Studies in Translatology, 16(1), 49–60.

Reinke, U. (2004) *Translation Memories: Systeme – Konzepte – Linguistische Optimierung*. Frankfurt: Peter Lang.

Ribas López, C. R. (2007) *Translation Memories As Vehicles For Error Propagation: A Pilot Study*, unpublished minor dissertation (M. A.), Universitat Rovira i Virgili, Tarragona.

Rieche, A. C. (2004) *Memória de tradução: auxílio ou empecilho?*, unpublished thesis (M. A.), Pontifícia Universidade Católica do Rio de Janeiro.

Somers, H. (2003) '*Translation Memory Systems' in Somers, H., ed., Computers and Translation: A Translator's Guide*. Amsterdam / Philadelphia: John Benjamin, 31-48.

A Communicative Approach to Evaluate Web Accessibility Localisation Using a Controlled Language Checker: the Case of Text Alternatives for Images

Silvia Rodríguez Vázquez[1,2], Jesús Torres del Rey[2]
[1] Multilingual Information Processing Department (TIM/ISSCO),
University of Geneva, Switzerland
[2] Department of Translation and Interpreting,
University of Salamanca, Spain
Silvia.Rodriguez@unige.ch; jtorres@usal.es

Abstract

As researchers in web localisation, web accessibility and controlled-language (CL) software, we have noticed a gap in the way the three can be brought together in mutually advantageous ways. Localisation in general has shown little awareness of accessibility matters. The different accessibility checkers that are available are seldom used by localisers and web accessibility regulatory bodies and assessment tools have traditionally been very vague about language-related requirements. In this paper, we explore a linguistic approach based on semiotics and communicative value to help localisers analyse the accessibility needs of web content (and, in particular, non-verbal elements such as images) by means of an evaluation methodology based on controlled-language rules integrated in a content authoring tool.

Keywords: *Web localisation, web accessibility, controlled language checker, images, text alternatives, accessibility evaluation, linguistic accessibility, semiotic analysis, communicative value*

1. Introduction

Broadband access to the World Wide Web and other internet services has very quickly become synonymous with having full citizenship in the 21st century. Texts, media, software, services, games: all commodities are converging into the Web (or the Cloud), which "has developed from a medium of information exchange and archiving in the academic community to the most commercially significant", the most influential global forum, "a mainstream mass communication medium" for all kinds of organisations and individuals (Boardman 2005, unit 1).

The Internet has created numerous new opportunities in the professional, academic, institutional, political, economic and social spheres, but in doing so, it has also opened a vast space of exclusion for those who have no access and are disconnected. As Tim Berners-Lee, W3C Director and inventor of the World Wide Web, has famously put it: "The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect" (http://www.w3.org/WAI/). Today, no one can be remotely up-to-date in any of the aforementioned areas without some regular quality access to the Internet.

It follows that when people cannot have access to it, and to the network of people, places, goods, knowledge and information that are instantly and ubiquitously offered, they are profoundly discriminated against. Therefore, we might argue that the most important barrier, besides personal or regional economics and language proficiency, has to do with physical and intellectual abilities, However, we prefer Harper and Yesilada's (2008, p.75) somewhat different diagnostics for web users with functional diversity: "People are disabled not by their impairment but are handicapped by the technology, infrastructure surrounding them, and the environment in which people are working in".

The World Wide Web Consortium (W3C) has made every effort to lower accessibility barriers. As early as 1997, the W3C launched the "Web Accessibility Initiative (WAI) to promote and achieve Web functionality for people with disabilities." They published the seminal Web Content Accessibility Guidelines (WCAG) as a Recommendation in 1999, and version 2.0 in 2008, with significant additions and redefinitions. This key document is structured around four principles of accessibility (webs must be

Perceivable, Operable, Understandable and Robust –the POUR principles), twelve guidelines to help implement these principles, and 61 different testable success criteria, so as to determine the degree to which each guideline is met (http://www.w3.org/TR/WCAG/).

In this context, what happens when a website needs to be made multilingual? What kinds of accessibility strategies or techniques will localisers implement, whether accessibility is explicitly included in the localisation brief, it is prescribed by national or regional law, it is identified by localisers as part of the intention, form or message in the "original" website, or it is felt as a professional duty? What tools are available to help them achieve target accessibility? Localisation in general and localisers in particular have shown little awareness of accessibility matters, probably because it has traditionally been considered beyond their "mild" technical prerogatives or capabilities, and mainly a developer's concern. The different accessibility checkers available are seldom used by these professionals, since most results seem to be of little relevance to localisers' mostly linguistic and macro-structural knowledge and expertise, or just a matter of poor *original* design. On the other hand, web accessibility regulatory bodies and assessment tools have usually been very vague about language-related requirements and have mainly focused on making sure alternative or simpler representations *exist* for components that can only be perceived, operated upon and understood by means of particular sensory capacities or intellectual conditions.

In this article, we will consider accessibilised web content as a kind of controlled language (CL), and webpages and websites as (hyper)texts comprising verbal and non-verbal communicative items, as well as language-dependent embedded applications. As Sharon O'Brien (2006, p.17) put it, the "objective of a CL is to improve the readability/comprehensibility of a text and/or its translatability". Since the relation between localisation and web accessibility has to do with localising controlled language, it makes sense to look at ways in which authoring and evaluation software based on CL rules can help the work of the professional localiser, in a similar way as QA or terminology checkers integrate with translators' toolboxes. A very positive collateral effect of this implementation would be to raise awareness about accessibility matters, which are quickly becoming a moral (and usually, legal) requirement for digital information.

2. Accessibility, Localisation and Language

Localisation and Accessibility have always been closely linked, if only because both activities aim at making a product accessible to a wider range of users than originally designed for. Like the former, the latter stands at the interface between a particular individual or user, a product or content to be used or processed, and the technology that makes that product possible and processable, both at the developer and the user ends (see Figure 1). To make something accessible ideally requires providing any users, irrespective of their abilities or according to their functional diversity, with a similar experience, or, at least, with a product that offers them equivalent value. Substitute linguistic variant for functional diversity (or language for ability) and we have the definition of localisation again. However, if we scratch the surface, differences start to emerge, as well as the need to redefine each on the basis of one another.

To start with, the "Web for all" main principles (http://www.w3.org/Consortium/mission.html) of the W3C include Social and Economic Development and Accessibility, but then also Internationalisation (not Localisation), "to make it possible to use Web technologies with different languages, scripts, and cultures" (http://www.w3.org/International/).

According to Pym (2001, p.1), Internationalisation "is the process of generalizing a product so that it can handle multiple languages and cultural conventions without the need for re-design". If we consider accessibility as "just another language" (Ó Broin 2004) —and we may add, *just another culture*, i.e. another set of conventions, usages, and interaction needs and habits to be taken into account, then we are talking about accessibility as synonymous with (or complementary to) internationalisation, as the process of at-*source* "neutralisation" of particular (technical, linguistic, cultural) traits for all languages and cultures. On the other hand, by turning around the equation of localisation as "a form of accessibility" (ibid.), the latter could be seen as the process of localising *into* particular accessible (*target*) languages. In short, accessibility means universalising, globalising, but also personalising, localising.

Accessibility, like internationalisation, can be part of the original design, or it can be a later adjustment. It is generally recommended that, for designers, internationalisation should be a mindset and not an

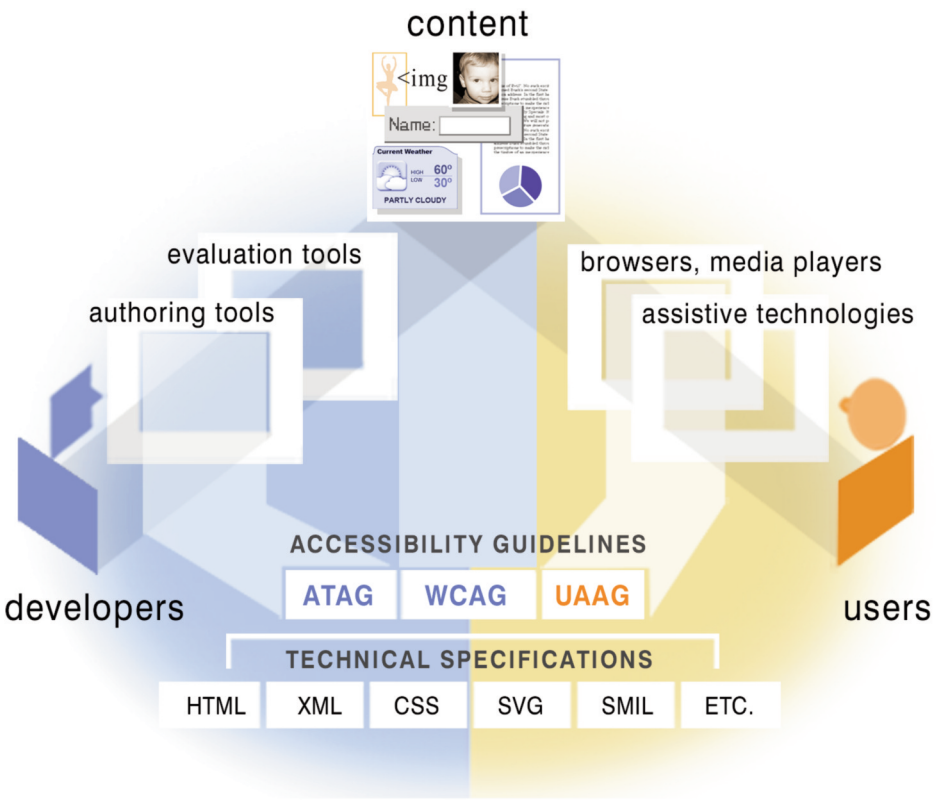

Figure 1: Essential Components (and Interaction) of Web Accessibility[1]

afterthought, in order to avoid as many problems as possible, as well as to make localisation smoother. This can be easily applied to accessibility, as that would mean, from a design perspective, to try to account for as many communities of users as possible in the product experience. From a localiser's perspective, a similar choice can be made between considering that accessibility must be *transferred* in parallel with content and, on the other hand, adopting a broader perspective or strategy whereby the product and its experience has to be made useable for as many as possible of the functionally-diverse target communities, thus localising with accessibility in (body and) mind.

As is well known in Translation Studies, a straight transfer ideology is problematic, since the message or content depends at least on context, shared or diverging expectations, intentions and knowledge, channel, form, and, most importantly, on what is implicit. If we combine it with the aforementioned recommendation for internationalisation —together with the idea that digital products are not just about content, but also (or mainly) about experiencing, doing and interacting— then we should conclude that internationalisation, accessibilisation or universalisation could never be achieved in full, since a technological product needs to communicate its potential use through verbal and non-verbal language, usage conventions, collective references to metaphors and to other cultural (thus culture-bound) products. Similarly, the way users interact with a technological product depends on shared codes (language) and assumptions, but also on the way their bodies and minds work. Functionally diverse users, therefore, need to build and share alternative or complementary codes and assumptions, based both on the "mainstream" ones and on the way they experience reality.

From the opposite perspective, technology pervades everything in a digital product, but technology is based on intelligence, which is also constructed through language. Now, language reflects how

[1] Image by Michael Duffy, from: Essential Components of Web Accessibility. S.L. Henry, ed. W3C (MIT, ERCIM, Keio). Status: Updated August 2005. www.w3.org/WAI/intro/components [accessed 5 Sept. 2012]

designers and code writers understand the world and interact with it, which, at the same time, depends on the way a language uses categories, names objects, builds and weaves relations through morphosyntax, lexical associations, semantics and pragmatics. Granted, many accessibility techniques are "just" embedded in the technology: for instance, the separation of content and layout through CSS; making sure all actions can be made via the keyboard; providing alternative descriptions to images; conforming code to specifications; making time limits or speed adjustable, and so on. However, most such techniques are dependent on language and communication as understood in our approach, and must be filled with actual representations and choices of content, layout, key conventions, informational structures, etc., which need to be recoded for different (or a comprehensive range of) user cultures. Even the 4.1 Robustness guideline, aimed at maximising compatibility with current and future user agents, has an important bearing on the localiser, not only because if a well-formed piece of code is broken in the process of localisation, the assistive technology might not be able to parse (Success Criterion 4.1.1.) and render the content to the user; but also because understanding the way standardised or localisable names, roles, values, properties or labels (Success Criterion 4.1.2) of specific web elements and components are used may be key to carrying meaning, intention and function across in a way that "assistive technologies can recognize and interact with"
(http://www.w3.org/TR/UNDERSTANDING-WCAG20/ensure-compat.html).

3. Localising Web Accessibility

3.1 Language-related web accessibility recommendations and techniques

With the purpose of assisting web authors, designers and developers in the promotion and implementation of accessibility in websites, the WAI introduced a set of sufficient and advisory techniques[2] to help meet three different accessibility levels (A, AA or AAA) of conformance with the success criteria in which the WCAG 2.0 guidelines have been divided up (see Figure 1). As the number of techniques introduced amounted to as many as 570, a series of subcategories were created so as to improve the usability of the document: General, HTML and XHTML, CSS, Client-side Scripting, Server-side Scripting, SMIL, Plain Text, ARIA, Flash, Silverlight and PDF Techniques. Although fairly complete and evidence-based, their universal and informative (not mandatory) nature has often distracted web professionals from taking a deep dive into the explanations, examples and essential recommendations linked to each guideline. As a consequence, these professionals have tended to favour alternative sets of techniques, equally acceptable, which were introduced by different relevant bodies or organizations, web accessibility stakeholders or their clients. What is more, authors have also ended up relying on their own judgment, or simply overlooking conformance to guidelines, finding them not relevant or too time-consuming to implement (Harper and Yesilada 2008).

As far as *language* accessibility guidance is concerned, WCAG 2.0 guidelines and success criteria often remain particularly abstract, which contrasts with the concreteness of the recommendations addressing more technical accessibility aspects. Similarly, through WCAG 2.0 Techniques, no specific semantic, syntactic or lexical hints are provided on how the final text should be written. The only advice given is very general in nature, thus inevitably making it subject to ample interpretation. Such techniques would include, for instance, using the clearest and simplest language appropriate for the content, clarifying the purpose of a link using the title element, or correctly describing the subject of the webpage in the text content of the <a> tag, while ensuring that it makes sense when read out of context (for example, by a screen reader or in a list of search results).

Another noticeable feature is that there is no subcategory fully addressing language-related issues (Plain Text Techniques embrace merely formatting conventions in accordance with Guideline 1.3 [Adaptable: Create content that can be presented in different ways without losing information or structure]). When present, language-related concerns are usually listed under General or HTML Technique groups, thus hampering their visibility and compliance by web authors. For example, Techniques G17 (indicate new content with boldface and a text indicator), H39 and H73 (use caption and summary elements to provide relevant information about tables), or G96 (ensure that items within a webpage are referenced in the content not only by shape, size, sound or location, but also in ways that do not depend on that sensory perception). On the

[2] "The 'sufficient techniques' are considered sufficient by the WCAG Working Group to meet the success criteria… 'Advisory techniques' can enhance accessibility, but did not qualify as sufficient techniques because they are not sufficient to meet the full requirements of the Success Criteria, they are not testable, and/or because they are good and effective techniques in some circumstances but not effective or helpful in others" (W3C, WAI 2008) http://www.w3.org/TR/WCAG20-TECHS/.

other hand, sufficient techniques usually cover, in more detail, methods for the accessibilisation of non-verbal elements of the website (e.g. colours, mark-up languages, videos…), whereas references to textual accessibility can often be found only under advisory techniques (for example, in the case of techniques for Guideline 3.1 [Make text content readable and understandable]). Thus, not surprisingly, cognitive, language and learning areas are normally highlighted as the weakest points in web accessibility assessments (Harper and Yesilada 2008; Access for all 2011).

This gap in the aforementioned accessibility areas has led researchers to look at related fields of study for answers. Many have underlined the potential of the use of the Semantic Web, stating that "it might enable typical Web content to be converted to a simplified, clearer or symbolic representation" (Seeman 2004, p.70), and arguing that "if a page's content is expressed through ontologies, this means that the application is able to manage this content and modify it, so that it can be shown in the most convenient way, following the guidelines for web accessibility" (Sevilla *et al* 2007, p.12). Similarly, Natural Language Processing (NLP) modules and web technologies have been combined in research studies in order to tackle obstacles faced by individuals with cognitive impairments or low literacy skills through syntactic and lexical simplification and elaboration, automatic summarisation, or name entities recognition and PoS —classification (Watanabe *et al* 2010). Nevertheless, approaches have usually been presented from a monolingual point of view, occasionally suggesting that "knowledge based accessibility techniques ubiquitously promote other aims of Web Design including device independence, internalization and localization" (Seeman 2004, p.68).

3.2 A framework to analyse localisation-related accessibility issues

As mentioned earlier, the accessibility transfer is hardly ever recognised as a fundamental step in the localisation process (or in the training of localisers, for that matter). Localisation professionals usually fail to prove the necessary accessibility know-how when adapting web products to the target audience, and to master the appropriate web accessibility evaluation tools. In fact, while results of an exploratory pilot study showed that analysing and improving the source webpage in terms of linguistic and stylistic accessibility before translation helped the localisation expert to achieve better readability results, localisers' accessibility knowledge also influenced the degree of language accessibility obtained in their respective target products (Rodríguez Vázquez and Torres del Rey 2011). This implies that, even if problems faced by people with cognitive, language and learning disabilities had been reduced in the source page, localised versions would still need to be assessed in terms of accessibility. So, how does a localiser go about checking whether a website is being properly accessibilised? Where does a localiser's attention need to focus to make sure accessibility is part of the medium and the message being projected onto the target locales (including, so to speak, target "sensoriales" or "functionales" [*ibid.*])? What elements of the web language can be controlled with the help of a combined authoring-evaluation tool?

In an inspiring recent article, Gutiérrez y Restrepo and Martínez Normand (2010) presented the WACG 2.0 requirements that, on the basis of their extensive work experience with web accessibility and technical translation, they believed to be most relevant for web content localisation. The localisation-related accessibility issues brought forward were organised around the four POUR principles and, ultimately, the success criteria associated with the twelve Web Content Accessibility Guidelines. However, no rationale was given as to how to come up with a consistent communicative framework to assess the degree of success. In this regard, as we have argued that web accessibilisation involves the use of a controlled language (CL), we might want to try to formalise the WCAG 2.0 success criteria into CL rules which take into account desired linguistic accessibility guidelines. Take, for instance, the categories and subcategories compiled by Sharon O'Brien (2003), who draws on Bloor and Bloor's criterion of primary functionality (what language area is influenced most: lexical, syntactic, textual and pragmatic rules). And yet, we are still missing a key aspect: many of the web components that a localiser must check for proper textual accessibilisation are non verbal, so before considering whether a certain rule must be followed, there must be a proper analysis of the communicative value of the items that need to be accessible. This is the gap we are trying to address.

We need a type of analysis that is based on a more comprehensive idea of language. After all, language-related accessibility techniques depend very much on users' diverse functionalities, but also on the way verbal and non-verbal (hyper)textual elements can

interact with each other and with the agents (assistive and/or human). All web elements can be considered to have linguistic structure (with subjects, agents, or actors; verbs or actions; objects acted upon; properties, etc.) and communicative value (there must be some sort of agreement as to what the web and the user can do at every moment). As Winograd and Flores argued (1986, p.176), communication "is not a process of transmitting information and symbols, but one of commitment and interpretation"; and digital objects (such as websites) are a "structured dynamic communication medium" that can represent and manipulate this "network of commitments" systematically. What is more, as technological objects, they are (like language) human extensions in the world (McLuhan & Powers 1995, p.24). Therefore, lexical, syntactic, textual and pragmatic rules must be extended to accommodate non-verbal communication and interaction.

This can be done by introducing a more semiotic framework, as understood, for instance, by Roland Barthes (1968). "Signs take the form of words, images, sounds, odours, flavours, acts or objects … Anything can be a sign as long as someone interprets it as 'signifying' something - referring to or standing for something other than itself. We interpret things as signs largely unconsciously by relating them to familiar systems of conventions. It is this meaningful use of signs which is at the heart of the concerns of semiotics" (Chandler 1999). In Barthes's view, verbal and non-verbal signs must all ultimately resort to the system and the process of language, where their signification is the result of the joint action of the signified (content) and the signifier (the material form, the designation, and the layout): "it appears increasingly more difficult to conceive a system of images and objects whose signifieds can exist independently of language: to perceive what a substance signifies is inevitably to fall back on the individuation of a language: there is no meaning which is not designated, and the world of signifieds is none other than that of language" (Barthes: introduction).

For the French semiotician (as for Widdowson), *value* (as opposed to "pure" signification) is the meaning of a sign i*n context, in relation with other signs*, when it is put to use for *communicative purposes*. It is this concept of value that we will use in order to assess the meaning and significance of the different signs in a webpage, and, crucially, as a benchmark for appraising success in localising accessibility. From a communicative, linguistic or semiotic point of view (we use these adjectives interchangeably as regards our approach, given their interrelationship), we might want to look into theories regarding speech acts, or any other accounts of non-referential uses of language, for a definition of the possible values of communicative items found in and around websites. To simplify, however, we suggest that attention should be focused on certain linguistic values of the content and layout of the different signs (roughly, computer code elements such as paragraph text, tables, images, hyperlinks, embedded video, and so on) present in a website: apart from *referential* meaning, it is important to assess the *functional, aesthetic* and *structural* value, which every sign irradiates (and is irradiated) about itself and the surrounding signs to a lower or higher degree through content and layout. In order to illustrate this approach, we will take the example of images (coded within the <img> tag), and particularly its *alt* attribute, which helps users determine what the non-verbal content is.

3.3 An example: text alternatives for images

The most common method to introduce short text alternatives[3] for images is providing an *alt* attribute within the HTML <img> element. Its main function is to serve as a substitute for the image in cases where the image itself cannot be displayed —for instance, when images are disabled through the web browser, while waiting for the images to download, when using text-only browsers...— or seen —for example, by users of screen readers or refreshable Braille devices with visual disabilities— (Craven 2006; WebAIM 2005). While the use of the *alt* attribute is explicitly recommended in Success Criteria 1.1.1 [All non-text content that is presented to the user has a text alternative that serves the equivalent purpose], under Guideline 1.1 [Provide text alternatives], there are multiple WCAG 2.0 Techniques covering the different usages of images and their text equivalent, either under the HTML subcategory or the General subcategory (see Table 1). Although they are not within the scope of this paper, it is worth mentioning that methods to provide descriptive information in other contexts are also present in various WCAG 2.0 Techniques; for example, G158 refers to the alternative text accompanying audio content and recommends the bracketed addition of "text transcript follows" or "text description follows" after the title of the file; and through G74, the WAI recommends to introduce a pertinent description of the non-verbal element as part of the standard

[3] When it is necessary to introduce a long-text description, W3C-WAI recommends the use of longdesc attributes (WCAG 2.0 Technique H45).

Image	*Sample scenario*	*WCAG 2.0 Techniques: Best practice for alt usage*	*Suggested accessible alt content*
Referential Value			
image complementing main textual content	A picture shows how a knot is tied including arrows showing how the ropes go to make the knot	G94: Convey same purpose and information as the non-text content	The text alternative describes how to tie the knot, not what the picture looks like.
ASCII art, emoticons and leetspeak	=8-0	H86: Offer a text explanation of what the picture is	"fright"
interactive area	Image depicting the floor plan of a building	H24: The *alt* serves the same purpose as the selectable regions of an image map	"Building's floor plan. Select a room for more information about the purpose or content of the room."
content image	Rating system in HTML: three filled stars and two empty stars	G196: Avoid unnecessary duplication that occurs when a grouping of adjacent non-text content is used to present information or functionality	First star: "3 out of 5 stars" Other four stars: "" [null *alt*][4]
Functional Value			
icon	A link contains text and icon, and the icon provides additional information about the target	H30: Use descriptive title for the link and add information about the target in the *alt*	"PDF format"
complex image, chart, graphic	Image/chart too complex	H45: Provide information in a separate file when a short text alternative does not adequately convey the function or information provided in the image	"a complex chart"
button	There are multiple submit buttons on a page, and each lead to different results.	H36: Using *alt* attributes on images used as submit buttons to indicate their specific function.	"submit form"
CAPTCHA	A CAPTCHA test asks the user to type in text that is displayed in an obscured image	G143: Provide a text alternative that describes the purpose of the CAPTCHA	"Type the word in the image"
Aesthetic & Structural Value			
decorative image	Image with an spiral introduced as a decorative element	H67: Mark images that should be ignored by Assistive Technology	null alt[5], or a definition of mood or aesthetics being transmitted
image representing unordered list	Bullet points used in a list as a visual formatting hint	Not explicitly considered under WCAG 2.0 Techniques. Recommendation by authors: either the list punctuation or an equivalent expression should be employed, at least if not obtrusive	"bullet point", "new item", "next item in the list"
general image, line	Line(s) dividing the webpage in different sections	Not explicitly considered under WCAG 2.0 Techniques. Use of CSS is usually recommended for this purpose. Recommendation by authors: Offer a text explanation section boundaries.	"End of section 1. Beginning of section 2".

Table 1: Classification of images (<img>) based on their communicative value and examples of recommended accessible alt content in English

[4] Except for the last two ones, sample scenarios have been taken from WCAG 2.0 Techniques.

[5] If the *alt* text is set to null (i.e. alt="" —recommended— or alt=" "), it means for assistive technology that the image can be safely ignored (W3C, WAI). Having a "null" *alt* attribute is not the same as having no *alt* attribute. While the former conveys a clear message to the user (it communicates), the latter, considered as a non-accessible technique, could lead users to think that they are missing important content in the page. Also, when the image has no *alt* attribute, some screen readers read the file name of the image, which can be confusing to listen to (WebAIM).

presentation, for instance, by locating it near the non-text content (e.g., "October sales chart for top three salespeople. Details in text following the chart: ").

The lack of accurate and standardised guidelines on how to apply short-text alternatives for images has resulted in many sets of recommendations published by different bodies, both at national and international level (WAI, WebAIM, US Access Board), and academics (like Slatin, Pilgrim, MacAlpine or Korpela). Contrary to what was expected, this has made what was a problematic situation into a chaotic one, triggering endless discussions on when to write the alternative texts and by whom, if they should exist or not depending on the type of image, which words to use, or how many total characters the wording should have (Craven 2006). However, in our research, based on the updated version of the WCAG 2.0 Techniques (dating from 3rd January 2012), we will apply a semiotic framework, as described in the previous subsection 3.2, to classify alternative texts on the basis of the communicative value of the corresponding image. We believe that this approach would enhance the language accessibility degree achieved both during web authoring and localisation processes, since the value of the message that needs to be transferred would be easier to retrieve and convey from a linguistic point of view. Besides, this might prove to be a potentially useful evaluation methodology for alternative text for images, which could be applied from a controlled language perspective, both at source and target levels.

Table 1 shows an example of a possible analysis of alternative texts for images based on the communicative value of the image as a sign within a webpage. Images are grouped according to the main (but not only) value of their content, position or layout. The last column offers a suggested text alternative according to the different communicative values perceived for the image:

4. Language-related web accessibility evaluation (WAE)

4.1 State of the art

4.1.1 General scenario

The achievement of linguistic accessibility in websites, regardless of or in accordance with their locale, is, as we have seen, a goal far from being met on a large scale nowadays. The vagueness of language-related techniques has influenced their evaluation, including the definition of linguistic patterns to be recommended or avoided. Generally speaking, "different methods exist to evaluate accessibility of web pages, which can be categorized into five groups: Inspection methods, automated testing, screening techniques, subjective assessment, and user testing" (Brajnik *et al* 2011, pp.249-250), although an even broader distinction is commonly made between automatic evaluation and human-based evaluation techniques. The former, despite the

Figure 2: www.tawdis.net web site checked by WebAIM's web accessibility evaluation tool WAVE

obvious advantages it has (higher volume of data processed, time savings, among others) —as it is "performed by software, without the need of direct human intervention, and with expertise embedded in a software framework/tool" (Fernandes and Carriço 2012, p.2)—, also presents important limitations, not only in terms of the depth and completeness of the analysis carried out (*idem*.), but also regarding the transparency of the results, since the successful production of error messages by the accessibility checking software often prevents the evaluator from knowing whether an important aspect has been omitted in the process.

provide a more detailed evaluation report, some tools have come up with solutions that are intuitive and offer more guidance to the user, such as classifying the errors per principle, guidelines, success criteria and techniques, or according to different typologies of error; in the case of IBM's web accessibility checker aDesigner, accessibility problems are labelled with colours which indicate, for instance, if the error has to be confirmed, or if it is an issue that needs a human check. And yet, techniques are just mentioned, but no further description about how to meet them is provided (see Figure 3: 42 items were listed under "Human check"; one of the remarks

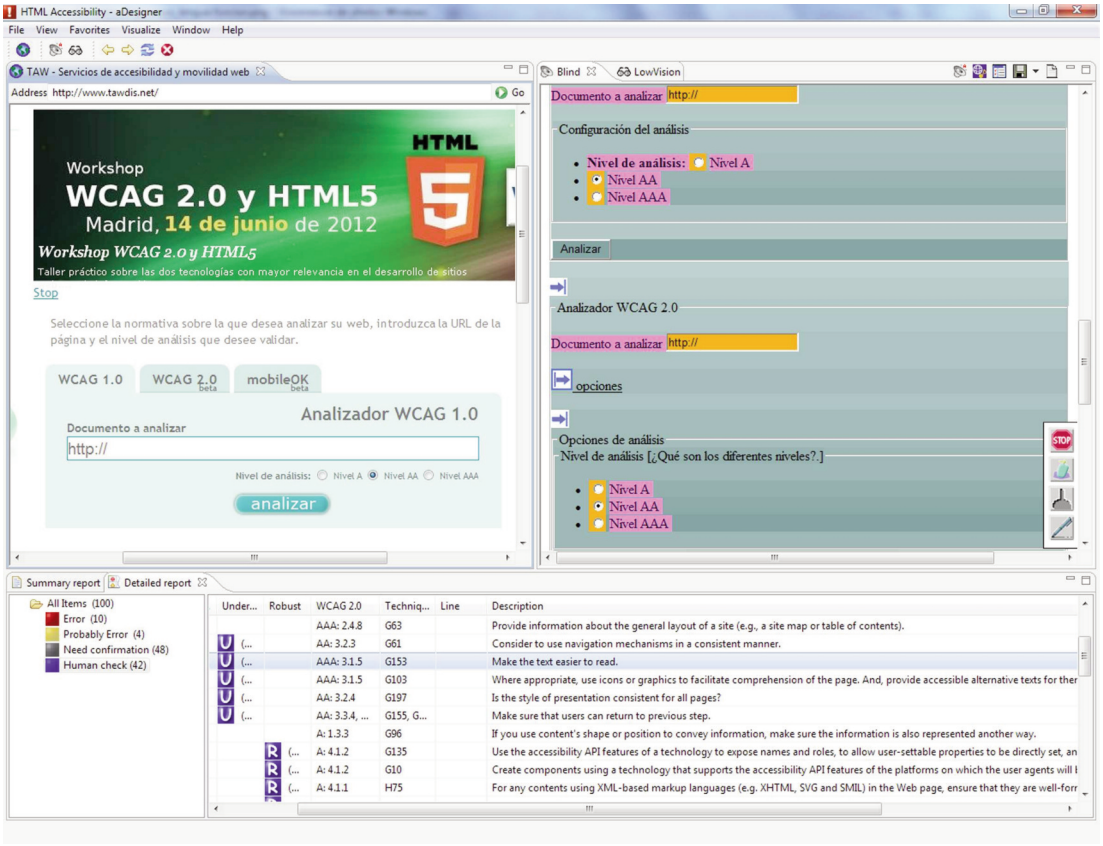

Figure 3: www.tawdis.net web site analysed by IBM's web accessibility checker aDesigner.

In Figure 2, for example, we can see that WAVE, the free web accessibility evaluation tool provided by WebAIM, has detected no accessibility errors. However, in the case of the message regarding the alternative text of an image (see arrow: "Feature: Linked image with alt text. Alternative text is present in an image that is a link"), the checker acknowledges the existence of the *alt* attribute but no information is given about its content. In order to

points to Technique G153 [Make the text easier to read]). It is finally worth mentioning that no direct references to specific linguistic issues (syntactic, stylistic, grammatical or lexical) as regards the language of the page (in this case, Spanish) are made in any of the examples from the figures.

All of these warnings usually create unnecessary noise, since they are given when "it is not possible to identify certain characteristic of an element as right

or wrong, without the need of an expert intervention" (Fernandes and Carriço 2012, p.2), and, sometimes, they point at accessibility issues that might already be solved. Evaluation verdicts in these cases very much depend on the metrics applied[6]. In order to clarify these warnings, simulations, subjective assessments and user testing sessions are usually carried out by a selection of people with functional diversity or by experts, who "can be characterized in terms of (a) the practice in using a specific evaluation method… and (b) the knowledge, practice, and skill in accessibility in general (on assistive technologies, typical accessibility problems, user behaviours or user preferences)" (Brajnik *et al* 2011, p.251). Nonetheless, what is known as the "evaluator effect" phenomenon still causes discrepancies regarding the existence of accessibility barriers or how severe they are (*idem.*). Although efforts have been made to create a Unified Web Evaluation Methodology (UWEM, http://www.wabcluster.org/uwem1_2/), covering also methods for manual content selection and interpretation of test results, the current version (1.2) has not been updated and still does not cover WCAG 2.0 Guidelines and Techniques.

In this regard, language-related accessibility testing guidance has often been consigned to oblivion, probably due to the factors that we have presented earlier. Incipient research is being carried out regarding syntactic simplification, and more broadly speaking, text adaptation to specific users, generally following two main operations: "remove unnecessary information from the text, and add information that better explains difficult terms" (Watanabe *et al* 2010). However, up to the present time, no communicative approach has been applied, multilingualism has not been taken into account, and only the needs of particular groups of users (people with low-literacy skills, people with dyslexia…) have been addressed. There is no doubt that final users should be involved in accessibility evaluations, but as deduced from different studies, the level of expertise of the evaluators plays a fundamental role in the quality of web accessibility assessments, as well as the metrics applied (Brajnik *et al* 2011). Taking this assertion into account, localisers should appear as the appropriate actors to validate linguistic accessibility issues, given their interdisciplinary knowledge, covering linguistic, cultural and technical web aspects. For instance, consistency and coherence issues (e.g. Technique H2, aiming at avoiding unnecessary duplication that occurs when adjacent text and iconic versions of a link are contained in a document) might go unnoticed to end users of accessible websites, contrary to what would be expected from localisation professionals.

4.1.2 Evaluating accessible text alternatives for images

As in the case of web textual content evaluation, assessment of text alternatives for images also relies on the subjectivity of the users performing the accessibility test. Besides, although the detection of *alt* attributes can be fully automated nowadays (see Figure 2), no deeper analysis of its content is featured in regular automatic web accessibility validators. Both facts, as well as authors' usual lack of awareness and advanced knowledge about the subject and the use of publishing software that automatically assigns text alternatives to images, have led us to consider the non-existence of good quality *alt* content as one of the main barriers identified throughout web accessibility studies (Access for all 2011). In order to bridge this gap, several automated checking methods have been introduced based on optical character recognition (OCR) techniques (analysing whether there is textual information in the image and whether it corresponds to the *alt* content), classification algorithms (such as Nearest Neighbor and Naïve Bayes), statistical data extraction, and dictionary-based word search (Bigham 2007). Other techniques have included comparative analysis of *alt* text length and image file size, or *alt* text length and number of images on the page (Craven 2006), as well as pattern recognition approaches. The latter have mainly shed light on elements that should not be present in the value of the attribute; for instance, non alphabet characters, file type abbreviations, HTML code or a continued series of numbers (e.g. alt="0111243.gif") (Goodwin 2010). Yet, already "many screen reader users write custom scripts in their screen reading software that prevents alternative text known to be bad, such as 'image', 'spacer' or '*' from being spoken" (Bigham 2007, p.349). Despite this significant progress, achieved in research projects still under development, a more language-focused approach is needed to reduce meaningless noise in automatic check results and provide specific linguistic-oriented guidelines.

4.2 Linguistic accessibility evaluation proposal

Machine-verifiable accessibility checkpoints are

[6] Fernandes and Carriço (2012), for instance, have used specific metrics in their experimental studies including three different rates: a Conservative rate (where 'warn' results are interpreted as failure), Optimistic rate (where 'warn' results are interpreted as passed) and Strict rate (where 'warn' results are dismissed). Depending on the rate chosen, a given webpage might be considered more or less accessible.

similar in number to those that cannot be validated through software (Vigo *et al* cited in Lopes and Carriço 2010). We are particularly concerned with bridging this gap with regard to language based accessibility issues. Our ultimate goal is to develop a methodology through which expert and automated evaluation practices are merged by means of a state-of-the-art controlled language checker, such as Acrolinx IQ, which would allow us to apply a non universal, customisable evaluation method, depending on the locale, the product, or even the end user's group(s). The first evaluation level of *alt* text (simple detection of *alt* attribute presence or omission within the <img> element) is already covered, as we have seen, by most accessibility validators. Our proposal, however, aims at reaching a second level of analysis, by providing users of accessibility checkers (whether web creators or localisers) with pertinent and valuable feedback on the linguistic accessibility quality that has been achieved, given a set of human-oriented controlled language (HOCL)-based rules covering lexical, grammatical and stylistic characteristics of accessible texts (for instance, avoidance of double negatives, length of sentences, degree and type of subordination, etc.). In this regard, and at this very same second level, spelling and grammar checks can also be run automatically, according to the locale of the page.

Yet, the potential of using such a tool for linguistic accessibility validation goes even further. Through the Acrolinx IQ Batch Checker, for instance, it is possible to reach a third level of analysis, and define the elements of the web document that we want to evaluate. Context Segmentation Definition (CSD) files determine the document segmentation settings, indicating which part or element (or sign, in our framework) of the web content will be checked against a certain group of rules. Based on this functionality, we propose an approach both for 1) achieving a more linguistically accurate evaluation of text alternatives based on their web context, and 2) guiding the localiser on how to accessibilise image-based content from a different perspective, that is, providing them with controlled language patterns and pragmatic information about the semiotic value of the *alt* attributes in the web page being localised.

The latter would constitute a fourth level of communicative analysis. After filtering (through CSDs) what signs (e.g. images) and subsigns (e.g. interactive areas or image maps, or graphic submit buttons) to validate (the above-mentioned third level of context- or sign- based analysis), the tool would provide the localiser with:

1) Hints about their semiotic value (e.g. Text alternatives for image maps are often descriptive, but also need to convey an instructive message at the end, so that users know how to interact with them);
2) The linguistic patterns (formalised in rules) often used or recommended when trying to communicate that specific value (e.g. noun phrases should be used for the first descriptive part, whereas the imperative form of certain verbs should be used in order to indicate orders);
3) Accessible and non-accessible examples of that category (see again suggested accessible *alt* content for interactive areas in Table 1).

In order to define those patterns, we need to analyse, beforehand, how the communicative value of images can be linguistically formalised, and to create context-based rules, to assign them to a given CSD. Feedback to the user would be provided through "negative accessibility indicators" (following O'Brien and Roturier's terminology, 2007) but also through positive guidance. This means, on the one hand, that the tool will not only look for accessibility problems, but it will also present the specific errors spotted and possible suggestions (if any) to correct them. On the other hand, it will show the text that has been validated, and offer an explanation of the linguistic patterns and associated communicative strategies that may have been used originally to accessibilise the web element, thus providing the localiser with important information for the task at hand.

Another important benefit of implementing this use of Acrolinx IQ, especially within the localisation process, would be the high level of customisation of the evaluation patterns depending on the language or locale. Since linguistic rules (and eventually the values of non-verbal signs[7]) are not always transferable from one locale to another, validation results would be more pertinent and reliable. We could even expand the tool's functionality to allow it to compare the language accessibility rules followed in each locale version of the webpage, which would

[7] Take, for instance, icon images such as ticks (check-marks), country flags, crosses or culture related mailboxes. Their meaning and semiotic value in the web might vary depending on the culture, the web product or the target users.

provide the localiser with valuable information about what was accessible in the source page, what needs to be maintained and what requires being adapted to the target locale. Taking all of the above into account, more CSD could be developed to assess the linguistic accessibility of other web elements containing text, such as <summary>, <caption>, <blockquote> and <legend>, or attributes like *title* or *longdesc*. Prior analysis from our proposed communicative approach would help to determine their linguistic characterisation, and applied rules could be created for a new controlled-language check.

5. Conclusions and future work

In this article we have brought to the forefront the idea that language is a key aspect in web accessibility. This definition of accessibility, which shares a lot of common ground with localisation, as underlined by Ultan Ó Broin (2004) and Gutiérrez y Restrepo (2010), has inspired our research, leading us to the proposal of a new theoretical communicative framework aimed at designing an evaluation methodology that could be helpful for web authors and for localisers. On the other hand, complementing existing WAE tools with state-of-the-art NLP-based software, such as Acrolinx IQ, could mean a significant improvement on the current degree of linguistic validation offered by web accessibility evaluation technology. From a localisation perspective, professionals in the field could also leverage the advantages of a language-based accessibility validator, either as a quality assurance technology or as a complementary tool to compensate for the lack of advanced knowledge in the matter.

The proposed theoretical framework has not yet been fully tested empirically, although there are indicators from the research and the industry communities that this path is worth pursuing. Pilot studies on linguistic accessibility validation have been successfully carried out with Acrolinx IQ up to the second level of analysis described at the beginning of subsection 4.2. We are currently analysing the data regarding image text alternatives extracted from an accessibility study on 100 Swiss pages (Access for all 2011) and we expect to have some positive results soon. We also intend to evaluate those outcomes involving both end users of accessible web pages and localisers. Although it still is an incipient work, our proposal offers the potential of interconnecting the fields of web accessibility, localisation, and NLP in a unique way that we believe will have an important impact on our research community.

References

Access for all (2011) *Étude 2011 sur l'accessibilité des sites Web suisses*, available: http://bit.ly/MZ6Ls0 [accessed 28 Jun 2012].

Barthes, Roland (1968 [1964]) *Elements of Semiology* (Trans.: Annette Lavers and Colin Smith), New York: Hill and Wang.

Boardman, M. (2005) *The Language of Websites* (Intertext Series), London & New York: Routledge [Kindle Edition].

Brajnik, G., Yesilada, Y., and Harper, S., (2011) 'The Expertise Effect on Web Accessibility Evaluation Methods', *Human–Computer Interaction*, 26, 246–283.

Bigham, J. (2007) 'Increasing Web Accessibility by Automatically Judging Alternative Text Quality', in *IUI'07 Proceedings of the 12th International Conference on Intelligent User Interfaces, Honolulu, 28-31 Jan 2007*, New York: ACM New York, 359-352 [online], available: http://bit.ly/OL9VFs [accessed 28 Jun 2012].

Chandler, D. (1999) *Semiotics for Beginners*, available: http://www.aber.ac.uk/media/Documents/S4B/semiotic.html [accessed 28 Jun 2012].

Craven, T. (2006) 'Some features of *alt* texts associated with images in Web pages', *Information Research* [online] 11(2), available: http://bit.ly/NVv4LY [accessed 2 Jun 2012].

Fernandes, N. and Carriço, L. (2012) 'A macroscopic Web accessibility evaluation at different processing phases', in *Proceedings of the 2012 International Cross-Disciplinary Conference on Web Accessibility (W4A)*, Lyon, 16-17 Apr 2012, New York: ACM New York, Article n.18 [online], available: http://bit.ly/Kwu29n [accessed 2 Jun 2012].

Goodwin Olsen, M., Snaprud, M. and Nietzio, A. (2010) 'Automatic Checking of Alternative Texts on Web Pages' in Miesenberger, K., Klaus, J., Zagler, W. and Karshmer, A., eds., *12th International Conference on Computers Helping People with Special Needs (ICCHP)*, Berlin: Springer, 425-531.

Gutiérrez y Restrepo, E. and Martínez Normand, L. (2010) 'Localisation and web accessibility', *Revista Tradumàtica 08* [online], Dec 2010, available: http://bit.ly/L3jhXz [accessed 2 Jun 2012].

Harper, S. and Yesilada, Y. (2008) 'Web Accessibility and guidelines' in Harper, S. and Yesilada, Y., eds., *Web Accessibility: A foundation for research*, London: Springer, 61-78.

McLuhan, Marshall and Powers, B. R. (1995[1989]) *La aldea global* (Trans.: Claudia Ferrari), Barcelona: Gedisa.

Lopes, R. and Carriço, L. (2010) 'Macroscopic characterisations on Web accessibility', *New Review of Hypermedia and Multimedia,* 16(3), 221-243, First published on: 17 November 2010 (iFirst).

O'Brien, S. (2003) 'Controlling Controlled English: An analysis of several controlled language rule sets', in *Proceedings of EAMT-CLAW-03*, Dublin, Ireland, 15-17 May 2003, 105-114 [online], available: http://bit.ly/14yupm [accessed 4 Jun 2012].

O'Brien, S. (2006) 'Controlled Language and Post-Editing', Writing for Translation (Getting Started Guide: October-November 2006). *Multilingual Computing and Technology #83*, 17(7), 17-19.

O'Brien, S. and Roturier, J. (2007) 'How Portable are Controlled Languages Rules? A Comparison of Two Empirical MT Studies', in *Proceedings of the XI MT Summit*, Copenhagen, Denmark, 10-14 Sept 2007, [online], 345-352, available: http://bit.ly/NzTh8I [accessed 3 Jun 2012].

Ó Broin, U. (2004) 'Accessibility is just another language', *Multilingual Computing and Technology* [online], 15(3), 17-20, available: http://bit.ly/M7xhlP [accessed 2 Jun 2012].

Rodríguez Vázquez, S. and Torres del Rey, J. (2011) 'Making digital content accessible: a collaborative task between the developer and the localizer', paper presented at *I International T3L Conference: Tradumatica, Translation Technologies & Localization: The coming of age of Translation Technologies in Translation Studies*, Universitat Autònoma de Barcelona, 21-22 Jun 2011.

Sevilla, J., Herrera, G., Martínez, B. and Alcantud, F. (2007) 'Web accessibility for Individuals with Cognitive Deficits: A Comparative Study between an Existing Commercial Web and Its Cognitively Accessible Equivalent', *ACM Transactions on Computer-Human Interaction (TOCHI)* [online] 14(3), 12/1-12/25, available: http://bit.ly/M7vJZf [accessed 2 Jun 2012].

Seeman, L. (2004) 'The Semantic Web, Web Accessibility, and Device Independence', in *Proceedings of the 2004 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*, New York, 18 May 2004, New York: ACM New York, 67-73 [online], available: http://bit.ly/L3pzVx [accessed 2 Jun 2012].

W3C Web Accessibility Initiative (2008) *How to Meet WCAG 2.0* [online], available: http://www.w3.org/WAI/WCAG20/quickref/ [accessed 3 Jun 2012].

Watanabe, W.M., Candido Jr., A., A. Amâncio, M., de Oliveira, M., A. S. Pardo, T., P. M. Fortes, R. and M. Aluísio, S. (2010) 'Adapting web content for low-literacy readers by using lexical elaboration and named entities labeling', in *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*, Raleigh, NC, 26 27 Apr 2010, New York: ACM New York, Article n.18 [online], available: http://bit.ly/Ka3h4R [accessed 2 Jun 2012].

WebAIM [Web Accessibility in Mind] (2005) 'The WebAIM Guide to Web Accessibility. Techniques and Concepts.' [CD ROM], 2005, 2 CD's.

Widdowson, H.G. (1978) *Teaching Language as Communication*, Oxford: Oxford University Press.

Winograd, T. and Flores F. (1986) *Understanding Computers and Cognition. A New Foundation for Design*, Norwook, New Jersey: Ablex.

Localisation Issues of Software Shortcut Keys

Dr. Gintautas Grigas, Dr. Tatjana Jevsikova, Agnė Strelkauskytė
Vilnius University Institute of Mathematics and Informatics,
Vilnius,
Lithuania
www.mii.vu.lt
gintautas.grigas@mii.vu.lt, tatjana.jevsikova@mii.vu.lt, agne.strelkauskyte@gmail.com

Abstract

No common agreement exists on whether all shortcut keys should be localised during software localisation or not. The main argument in favour of the localisation of shortcut keys is a possibility to preserve mnemonics in the target language, whereas the main argument against their localisation is a possibility to maintain uniform command letters in the source and target languages. The aim of this paper is to investigate the localisation of shortcut keys and find a compromise between the contradictions mentioned above. The statistics of letters used in the command keys (i.e., Ctrl+letter) of 50 popular computer programs have been collected and analysed. The stability of command-letter pairing among different programs is evaluated and the recommendations for localisers are presented. The recommendations are based on the existing traditions of software design, existing practices of the major software producers, and the stability of command-letter pairing. The letters of command keys are divided into three main categories according to the strength of the relationship between the command and the letter. The categories are as follows: international (not to be localised), those that may be localised, and those that should be localised. Features of the combinations of the command keys with numbers and special characters are discussed as well. One more finding is that the Ctrl+Alt combination in the source program must be considered as an internationalisation error, since almost all languages that use the Latin script have letter keys with some characters on the third keyboard level, and the aforementioned key combination is equivalent to Alt Gr which is the recognised key to access third level characters.

Keywords: *shortcut key localisation, access key localisation, command key localisation, software localisation, software internationalisation.*

1. Introduction

Shortcut keys are used to provide an alternate and, usually, quicker way of navigating and invoking the operations of software programs. Different manufacturers name shortcut keys differently, e.g. shortcuts or hotkeys. The two main types of shortcut keys, in most software applications, are command keys and access keys.

A command key is a keyboard key (e.g. *F1*, *Home*) or a finite combination of keyboard keys (e.g. *Ctrl+C*) that is used to invoke a command associated with the key. The activity scope of command keys is a part of a program, an entire program, or a whole platform, e.g. an operating system.

Access keys are usually made up by combining the *Alt* key with a single letter, e.g., *Alt+F*, *Alt+Ž*. They are used to provide an alternate method to access drop-down menu functions of the software. They either open a lower level menu or execute a command, if there is no lower level menu. Access keys are only valid in a limited context, i.e., usually in a certain level of a drop-down menu or in a dialog box view where the appropriate command name is used.

The letter, used in an access key combination, is called a mnemonic letter. It usually reflects the name of a command. The mnemonic letters usually appear as underlined letters in the associated command names, for instance, *File*, *Edit*, and *Help* on a menu bar. These letters are always localised. The question of whether one of these letters should be localised or not does not arise since the letter to be underlined must be included in the localised (translated) command name. Otherwise they are referred to separately, e.g. *Datei F* for *File* command in German localisation. It is natural that the word, which is translated to the target language, may not have the same letter to underline as the original word. However, command keys are not localised in many cases. Thus, this article focuses on the problems of

localising command keys only.

Command key combinations are usually obtained by pressing a modifier key, marked with *Ctrl* (*Strg* in the German keyboard, *Vald* in the Lithuanian keyboard, etc.), or with the ⌘ sign on Apple computer keyboards. This paper will use the English abbreviation *Ctrl* to refer to this kind of a modifier key.

The validity scope of command keys is broader than that of access keys. Command keys present more universal functions compared to access keys. Therefore, the appropriate letters in command key combinations during localisation have to be selected with great care. Naturally, the letter should resemble the command in some way, e.g. be the same as the first letter of the command name, or have another relationship with the command operation, e.g. the letter *X* in the key of *Cut* command *Ctrl+X* resembles scissors, meanwhile the plus sign is suitable for the *Zoom in* command key (*Ctrl*++), and minus is suitable for the command key *Zoom out* (*Ctrl*+–).

English version		Localised version (Lithuanian example)	
Replace	Ctrl+R	Pakeisti	Vald+R
Edit	Ctrl+E	Taisyti	Vald+E
Select All	Ctrl+A	Žymėti viską	Vald+A

Table 1: The discrepancies between localised access keys and command keys letters

The shortcut keys use the keyboard level principle to add extra functionality to existing keys. As not all of these levels are used throughout different locales, we will explain the concept of the keyboard levels in more detail. A keyboard key, located in the alphanumeric part of the keyboard, may have one, two or more characters associated with it. The more characters a locale uses, the more characters are associated with the key.

Keyboard level 1 is a keyboard state that is accessed when the first character, associated with the key is entered by pressing that key (usually, a lower case letter of the alphabet). No modifier key is used to access level 1 of the keyboard. Level 2 is a keyboard state that is accessed when the second character, associated with the key is entered (usually, a capital (uppercase) letter or a number in some locales). Level 2 is activated when *Shift* key is pressed.

Finally, level 3 is a keyboard state that allows the third character, associated with the key to be entered. Level 3 is activated by holding the *Alt Gr* key and is used in many locales (German, French, Polish, Lithuanian, etc.) to enter special characters as quotation marks, the euro sign, mathematical symbols, etc. More details on the level 3 characters in different locales will be given in Table 6.

There is no consensus on the localisation of command keys. Hall and Hudson (1997) state that command keys should be localised because the letters should be meaningful, i.e. reflect the name of a command the key invokes. For example, in the English user interface, the key of the command *Print* should use the first letter of the command name (*Ctrl+P*). Whereas in other languages the letter *P* carries no meaningful information about the printing command since the name of the localised command starts with a different letter, e.g. *Drucken* in German, *Tulosta* in Finnish, and *Stampa* in Italian. A similar situation arises with many other commands. In Lithuanian, out of 26 English letters, only one command key *Vald+N* (*Ctrl+N*) has meaningful information: *N – Naujas* (*New*) (Grigas and Strelkauskytė 2011).

The discrepancies between localised and non-localised command key letters are clearly visible when command names, access keys and command keys are displayed side by side on the menu bar (Table 1).

Müler (2009) considers shortcuts and hotkeys as important elements of the user interface, to be internationalised and localised, that have to be included in test tasks of the user interface. The scholar does not observe any exceptional rules for the localisation of command keys: “Typically test tasks ensure that each UI element is internationalized and localizable, including menus, field labels, buttons, tooltips, hotkeys, shortcuts, messages, combo boxes and icons” (Müler 2009, p. 14). There are recommendations to distinguish mnemonics from keyboard shortcuts and not to localise keyboard

shortcuts. "Mnemonics are distinguishable from keyboard shortcuts. One difference between them is that the keyboard shortcuts are not localized on multi-language software but the mnemonics are generally localized to reflect the symbols and letters used in the specific locale" (Keyboard Shortcut 2012). Safar and Machala emphasise the importance of conformity of localisation in different software: "Ideally, existing shortcuts should not change between releases and should have a high degree of conformity with the operating system or similar and related applications" (Safar and Machala 2010, p. 3). Esselink (2000, p. 72) states that command keys may be localised, but special attention needs to be paid to special characters (@ $ { } [ ] \ ~ | ^ ‘ <>), which can cause problems when a non-English keyboard layout is used (the author was referring to the U.S. keyboard here and it should be noted that these characters can cause problems in the keyboard layout of the United Kingdom as well). During the development of international software Esselink (p. 34) also recommends using function keys instead of letters in shortcut key combinations (e.g. *Ctrl + F3*).

One of the *Microsoft Office Word Help* documents states: "The shortcut keys described in this Help topic refer to the U.S. keyboard layout. Keys on other layouts may not correspond exactly to the keys on a U.S. keyboard" (Microsoft Corporation 2007).

As there is no consensus on the question of whether command keys should be localised or not, the command keys are not usually localised. The main reason for this is a statement that letters, corresponding to the commands, are well established, well-known, and, therefore, should not be localised. Not all software developers provide the option of changing letters during localisation. If such an option exists, it is usually recommended not to change the letters.

Letters that are often used for command keys in the original (English) software versions are easily memorised. Usually this is the first letter of the name of the command. This principle is not retained in the localised software and the statement that "the localised software should look and feel as if it has been developed in the target culture" (Schäler 2003) is ignored. Therefore, the question of how to reconcile such contradictions appears.

The versatile way is to perform a deeper examination of the situation and find a compromise. The letters that have been strongly "tied" to the commands may be left as non-localised, whereas other letters should be localised, i.e. changed to preserve the relationship with the localised command name in the target language. This paper aims to analyse a permanence of pairs of commands and single letters that are frequently used in software applications, and provide recommendations for international software developers and localisers on the basis of these findings. Original (non-localised) applications have been selected for the analysis for two reasons. Firstly, the process of localisation starts with the original software. Secondly, the data used from the localised software may not objectively reflect the situation due to the aforementioned arguments towards the non-localisation of shortcut keys. The purpose and mnemonics of numbers and special characters do not significantly differ in various languages. Thus, they are not analysed in this paper. However the layout of these characters differs in various languages and raises some localisation issues. Numbers on the U.S. English keyboard and many other keyboards are located on the first level (case). However in French, Belgian, and Lithuanian keyboards numbers are located on the second level. So the command key *Ctrl+3* on US English keyboard becomes *Ctrl+Shift+3* on the French keyboard.

Another problem relates to the special characters located on the third level. A US English keyboard has two levels. Other locales usually have three levels. On the U.S. keyboard layout all special characters are available on the first and second levels of the keyboard, while many of them are located on the third level on the keyboards used in other countries. Third level characters are obtained by pressing the *Alt Gr* key together with the corresponding character key, e.g. *Alt Gr* + @. Conflict arises when shortcut key *Alt Gr* + @ is used in English software. A similar conflicting situation arises when *Ctrl* + *Alt* (left) + *special character* is used as shortcut key because the key combination *Ctrl* + *Alt* (left) is used to model *Alt Gr* key for older keyboards without this key.

Inconsistency may also arise due to differences in grouping special characters on different keyboard layouts (e.g. a dot and a colon are located on the same key on the German keyboard but not on the English keyboard).

2. The Analysis of Shortcut *Ctrl+letter* Usage

The analysis of the letters on the command keys has been based on 50 frequently used software programs. Programs for various purposes and from various distribution methods (open source, proprietary, and

Modifier key Ctrl+	Command 1		Command 2		Command 3		Command 4		Command 5	
	Command	Number	Command	Number	Command	Number	Command	Number	Command	Number
A	Select All	39								
B	Bold	9	Bookmarks	5						
C	Copy	40								
D	Deselect	5	Add bookmark	4	Duplicate	4	Font	3	Add a favorite	2
E	Center	8	Search	5	Edit	3	Export	3	E-Mail	2
F	Find	30	Search	4	Filter	3	Repeat	2		
G	Go To	9	Find Again	6	Find	4	Group	3		
H	History	7	Replace	5	Hide	3	Size	3		
I	Italic	9	Invert	5	Import	4	Information	4	Search	2
J	Justify	8	Downloads	3	Jump to	2				
K	Hyperlink	4	Check	2	Preferences	2				
L	Left	8	Open	3	Address field	2	Full screen	2	Levels	2
M	Message	2	Merge	2	Formatting	2				
N	New	35	Create	3						
O	Open	38								
P	Print	34	Copy	2	Preferences	2				
Q	Quit	11	Exit	9	Quick view	4				
R	Right	8	Refresh	6	Replace	6	Reload	4	Resize	2
S	Save	37								
T	New Tab	8	Text	2						
U	Underline	8	Source	7						
V	Paste	40								
W	Close	31								
X	Cut	39								
Y	Redo	18	Repeat	3	Crop	2				
Z	Undo	33	Comment	2						

Table 2: Information on command keys, presented according to the letters

commercial) have been selected. Most of the selected programs run on the MS Windows operating system while a number of them operate on Linux and MacOS. The majority of the programs have been localised into other languages (the number of existing localisations is one of the most important indicators of the popularity of the software).

The selected programs differ in size. To reduce the difference large software packages such as Microsoft Office and OpenOffice.org have been split into separate components (e.g. text processor, spreadsheets, and presentation editor). The data for the analysis has been taken from existing surveys (*Table of keyboard shortcuts 2012*), as well as studying the programs' documentation. Other data has been collected by carrying out experiments with running programs. The results of the analysis are presented in two ways: according to the letters, and according to the commands.

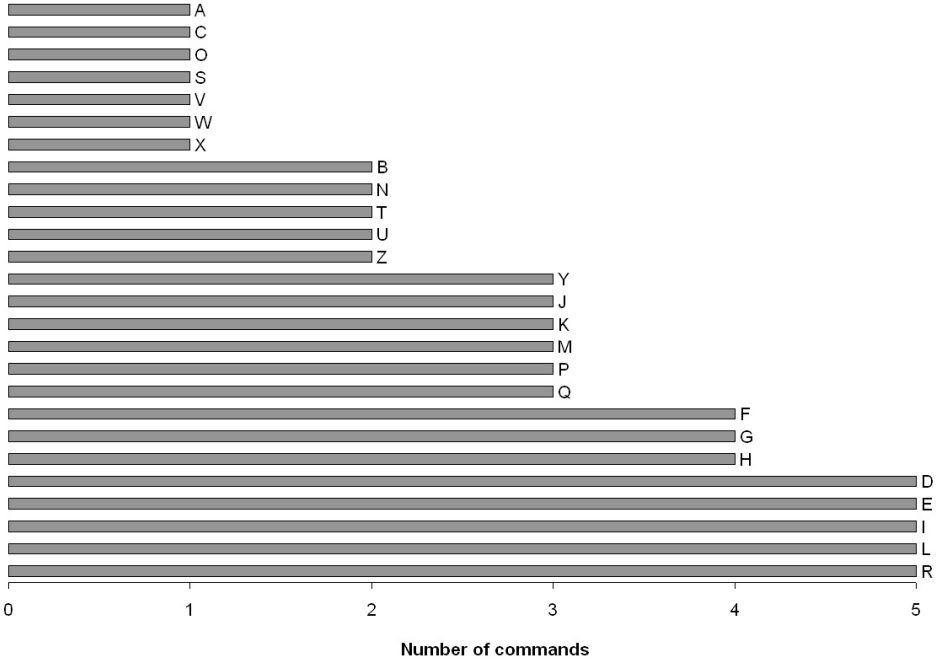

Figure 1: The number of programs, where the letter is associated with "the most popular" command

2.1 Results, presented according to the letters

Table 2 reveals information about the types of commands and in how many programs they correspond to each letter of the English alphabet. Five is the maximum number of commands that corresponds to the same letter, which has been observed in all the set of programs.

Each letter is presented by a row with five double columns for the commands using that letter. Each double column consists of two sub columns with the name of the command and the number of programs where the command is used. The first column depicts the largest number, which means that the command is more tightly associated with the corresponding letter.

Figure 1 presents the number of commands each letter is related to. This corresponds to the number of non-empty double columns in Table 2.

Letters associated with only one command may be considered as the ones that maintain a unique relationship with a single command. There are 7 such letters: *A, C, O, S, V, W*, and *X*.

Another important property is the number of programs where the letter is used (this feature is related to the column "Command 1" in Table 2, i.e. "the most popular") (Fig. 2). In fact the numbers come from Table 2, column "Command 1" and are presented graphically.

2.2 Results, presented according to the commands

Table 3 presents information about letters and

programs that are related to each command.

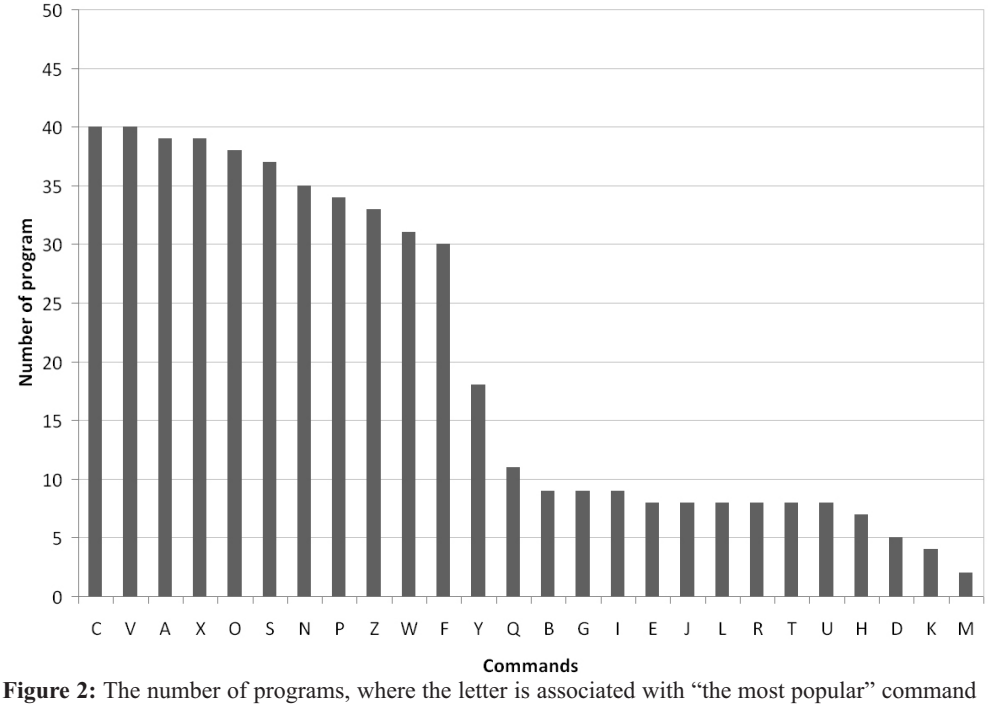

Figure 2: The number of programs, where the letter is associated with "the most popular" command

145 different commands were found in the 50 programs that were chosen for the analysis. It would be difficult to present all the data in a single table since such a table would consist of 50×145 cells. However, only the total numbers of commands are important for the analysis and not their distribution among the programs. Therefore, 14 frequently used programs, serving as an example set, have been included. The last column of the table provides information about the number of appropriate command keys in all other programs that have been examined and are not indicated in Table 3 (MS Office Power Point, Mozilla Thunderbird, Adobe Reader, etc.).

The commands, presented in Table 3 are sorted in descending order according to the number of programs that apply them. The larger the number, the stronger the relationship between the command and the letter is. The relationship is weaker if the same command is related to more than one letter. Four such commands are highlighted in Table 3: *Replace*, *Find*, *Search,* and *Open*.

One more factor that weakens the relationship between the letter and the command is the usage of the same letter for several different commands. In this case the priority is given to the command that is related to a larger number of letters. Such a relationship can be considered as the strongest one, while the relationship between all other commands with that letter can be considered as weak. The latter are highlighted after the commands with strong relationships in the grey shaded cells. The last command presented in the table, is *Message*. It is related to the final, and still unoccupied, letter M. After the letter M there are no commands marked by strong relationship.

2.3 Mnemonics of the letters

Data, presented in Table 3, can be analysed in terms of mnemonics of letters. Three commands, related to letters *X, V*, and *Z* do not bear direct relation to any of the letters contained in the English words such as *Cut*, *Paste*, and *Undo*. However, the visual representation of these letters reflects graphical icons of the commands, i.e., the letter *X* resembles scissors (the text or another object is cut); the letter *V* resembles an arrow pointing downward "into" the document to paste an object; whereas the letter *Z* signifies a zigzag, striking out a mistake. Therefore, the relationship between these three letters and their appropriate commands can be treated as an international (language independent) decision, and these command keys should not be localised. The

Command			Program														
Number of programs	Command name	Letter	Microsoft Word	Microsoft Excel	OpenOffice.org Write	OpenOffice.org Calc	AbiWord	Mozilla Firefox	Internet Explorer	SeaMonkey	Opera	Free Pascal	Gimp	Picasa	Skype	Total Commander	No of other programs
40	**C**opy	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	26
40	**Paste**	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	26
39	Select **A**ll	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	25
39	**Cut**	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	25
38	**O**pen	O	O	O	O	O	O	O	O	O	O	O	O	O			26
37	**S**ave	S	S	S	S	S	S	S	S	S	S	S	S	S			25
35	**N**ew	N	N	N	N	N	N	N	N	N	N	N	N	N		N	22
34	**P**rint	P	P	P	P	P	P	P	P	P	P		P	P			23
33	**Undo**	Z	Z	Z	Z	Z	Z	Z		Z	Z	Z	Z		Z		22
31	Close (**W**indow)	W	W	W	W	W	W	W	W	W	W		W			W	20
30	**F**ind	F	F	F	F	F	F	F	F	F	F	F			F		19
18	Redo	Y			Y	Y	Y	Y		Y	Y	Y	Y		Y		9
10	**Q**uit	Q					Q			Q			Q				7
9	**B**old	B	B	B	B	B											5
9	Exit	Q			Q	Q											7
9	**G**o to	G	G	G			G			G							5
9	**I**talic	I	I	I	I	I	I										4
8	Align **J**ustify	J	J		J	J	J								J		3
8	Align **L**eft	L	L		L	L	L										4
8	Align **R**ight	R	R		R	R	R								R		3
8	Center	E	E		E	E	E								E		3
8	New **T**ab	T						T	T		T					T	4
8	**U**nderline	U	U	U	U	U	U										3
7	**H**istory	H						H	H		H						4
7	Source	U						U		U	U					U	3
6	Find Again	G						G		G							4
6	**R**efresh	R							R								5
6	**R**eplace	R										R					5

Table 3 - Part 1: Information on command keys, presented according to the commands

5	**B**ookmarks	B						B		B	B						2
5	**D**eselect	D												D			4
5	Invert	I											I	I			3
5	Replace	H	H	H			H										2
5	Search	E						E	E								3
4	Add bookmark	D						D									3
4	Duplicate	D											D				3
4	Find	G															4
4	Import	I															4
4	Information	I								I							3
4	Hyperlink	K	K	K													2
4	Quick view	Q							Q							Q	2
4	Reload	R						R		R	R						1
4	Search	F												F			3
3	Create	N										N					2
3	Downloads	J						J			J						1
3	Edit	E								E							2
3	Export	E															3
3	Filter	F															3
3	Font	D	D				D										1
3	Group	G															3
3	Hide	H															3
3	Open	L						O									2
3	Repeat	Y	Y	Y													1
3	Size	H															3
2	**M**essage	M								M							1
2	**Number of other commands in 2 programs**		0	0	1	1	0	0	0	0	2	0	1	1	0	1	29
1	**Number of other commands in 1 program**		2	3	1	0	3	2	7	4	4	1	6	6	0	10	

Table 3 - Part 2: Information on command keys, presented according to the commands
Note. Bold font is used to indicate command names associated with international command keys (Paste, Copy, and Undo) and the first mnemonic letters of commands. Grey cells are used to indicate the repeated (in top – down direction) command names.

keys have been included in the analysis to provide a complete overview and confirm that these three commands are not related to any other letters.

Analysing the other 23 letters that have strong relationships with the commands, it has been observed that 19 letters have a mnemonic relationship with the English command names. On the one hand, this supports the localisation of the keys; on the other hand, the table shows that the

commands are strongly related to the letters. Both points are important. Therefore, the localisation should be carried out by means of preserving the same level of stability as in the original version of the software. This can be achieved by localising the command key letters in all programs. The larger the number of programs that use the command, the more difficult it becomes to achieve the goal. If the command is used in only one program, the key can be freely localised. There is also no considerable

Ctrl+	English	German	French	Spanish	Lithuanian	Finnish	Polish
A	Select **A**ll	**A**lles markieren	Sélectionner tout	Seleccionar todo	Pažymėti visus	Valitse kaikki	Zaznacz wszystko
B	**B**old	Fett	Gras	Negrita	Paryškintasis	Lihavointi	Pogrubiony
C	**C**opy	Kopieren	**C**opier	**C**opiar	Kopijuoti	Kopioida	Kopiować
D	**D**eselect	Auswahl aufheben	**D**ésélectionner	Anular selección	Naikinti pasirinkimą	Poista valinta	Anuluj wybór
E	Center	Zentriert	Centre	Centro	Centruoti	Keskittää	Wyśrodkować
F	**F**ind	Suchen	Rechercher	Buscar	Rasti	Etsi	Znajdź
G	**G**o To	Wechseln zu	Atteindre	Ir a	Eiti į	Siirry	Przejdź do
H	**H**istory	Verlauf	**H**istorique	**H**istorial	Retrospektyva	**H**istoria	**H**istoria
I	**I**talic	Kursiv	**I**talique	Cursiva	Pasvirasis	Kursivointi	Kursywa
J	**J**ustify	Im Blocksatz ausrichten	**J**ustifier	**J**ustificar	Abipusė lygiuotė	Tasata molemmatre unat	**J**ustować
K	Hyperlink	Link	Lien hypertexte	Hipervínculo	Saitas	Hyperlinkki	Hiperłącze
L	**L**eft	**L**inks	Gauche	Izquierdo	Kairė	Vasen	**L**ewe
M	**M**essage	Nachricht	**M**essage	**M**ensaje	Pranešimas	Viesti	Wiadomość
N	**N**ew	**N**eu	**N**ouveau	**N**uevo	**N**aujas	Uusi	**N**owy
O	**O**pen	Öffnen	**O**uvrir	Abrir	Atidaryti	Avoin	**O**twórz
P	**P**rint	Drucken	Imprimer	Imprimir	Spausdinti	Tulosta	Drukuj
Q	**Q**uit	Beenden	**Q**uitter	Salir	Baigti	Hiljainen	Zamknij
R	**R**ight	**R**echts	Droit	Derecho	Dešinė	Oikeus	Prawo
S	**S**ave	**S**peichern	Enregistrer	Guardar	Įrašyti	Tallenna	Zapisz
T	New **T**ab	Neue Registerkarte	Nouve longlet	Nueva pestaña	Nauja kortelė	Uusi välilehti	Nowa karta
U	**U**nderline	**U**nterstreichen	Souligné	Subrayado	Pabraukti	Alleviivattu	Podkreślenie
W	Close	Schließen	Fermer	Cerrar	Uždaryti	Sulje	Zamknij
Y	Redo	Wiederholen	Rétablir	Rehacer	Perdaryti	Tehdä uudelleen	Wykonaj ponownie
Number	19	6	9	5	1	1	5
%	83	26	39	22	4	4	22

Table 4: Command key letters in English software and their mnemonics properties in various languages
Note. The command names have been taken from the Microsoft Language Portal (Microsoft Corporation 2010).

difficulty if the command is used in 2 or 3 programs.

Table 4 presents the mnemonics situation for a number of languages from different groups (German, Roman, Baltic, Finno-Ugric, and Slavic) when command key letters were not localised. The table indicates that the situation in all languages is considerably less satisfactory when compared with English (non-mnemonic keys for localised command names are marked in grey background).

3. Recommendations for the Localisation of Shortcut Keys

The results of the analysis of command keys in relation to commands are less fragmented than those of the letters. One command is usually less likely to be related to several letters than one letter being related to several commands. This is obvious because the number of letters available in the alphabet is lesser than the number of commands. On the other hand, the relationship between the command and the letter is initiated with the command, i.e. a letter for a particular command is selected but not vice versa. The results are presented in Table 5.

These commands in Table 5 are presented in the same order as in Table 3, except for the fact that three commands, which have "international" status, are brought to the front of the list and marked with '1' in the row 2. In the other rows of Table 5, factors that have an impact on localisation are marked with '1'. They are the shortcut keys:

1 With letters that possess "international" status,
2 That are recommended to unify (*Keyboard Shortcuts* 2012),
3 With letters that are related to one command only (according to Fig. 1).

From this information, and the information in Table 3, the following recommendations about the localisation of command keys could be presented.

1. Command keys that have international status are not subject to localisation. Their commands have a mnemonic connection to the visual appearance of the letter and are not related to the usage of the command name in some languages.

2. The keys of commands that are not mentioned in Table 5 have to be localised. 119 such commands have been found in the analysed programs. Their keys are either rarely used (in one or two programs out of the 50 programs analysed), or their letters are used in other key combinations that are used more often. For these commands, the letters related to other commands should be used. Such relationships are unavoidable since there are fewer letters than commands. Even the letter *Z*, which is internationally accepted, is used for other commands in the original programs (see Fig. 1). However, it is important that the letter selected during localisation should not conflict with other command keys for the same program. This is a common rule for any shortcut keys (command keys and access keys) that are successfully implemented in software products.

No.	Command	Cut	Paste	Undo	Copy	Select All	Open	Save	New	Print	Close Window	Find	Redo	Quit	Bold	Go to	Italic	Align Justify	Align Left	Align Right	Center	New Tab	Underline	History	Duplicate	Hyperlink	Message
1	**Letter**	X	V	Z	C	A	O	S	N	P	W	F	Y	Q	B	G	I	J	L	R	E	T	U	H	D	K	M
2	**International**	1	1	1																							
3	**Recommended to unify**	1	1	1	1	1	1	1	1	1	1	1		1	1		1						1				
4	**Letter – 1 command**	1	1		1	1	1	1			1																

Table 5: Predominant pairs of letters and commands

3. It is more difficult to present recommendations for the other commands, listed in Table 5. The commands are presented in descending order according to both their usage in programs and in terms of localisation difficulties. If a key is localised, it should be localised in all other programs appropriately. The more programs that use the command, the more difficult it becomes

to use the same letter in all of the programs.

Additionally, the recommendation to unify command keys, i.e. to leave them all unlocalised, or to make them unified in the target locale should be taken into consideration. The attribute in the last row of Table 5 indicates that the letter is related to the same command in all programs, and it becomes the responsibility of the localisers to not reduce the number of the keys that possess the same quality.

4. Evaluating the attributes that are presented in rows 3 and 4 of Table 5 it becomes possible to modify the order of the letters (as well as one of the corresponding commands), presented in the first row of this table. The list of the letters below is presented starting with letters that have higher priority unification:

Keyboard layout	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Albanian		{		Đ		[	]				ł	Ł	§	}			\		đ			@	|			
Czech		{	&	Đ	€	[	]				ł	Ł		}			\		đ			@	|	#		
Danish					€								µ													
Dutch			¢		€								µ					¶	ß					»		«
Estonian					€														Š							Ž
Finnish					€								µ													
French					€																					
German					€								µ				@									
Hungarian	ä	{	&	Đ	Ä	[	]		Í	í	ł	Ł	<	}			\		đ		€	@	|	#		>
Irish	Á				É				Í						Ó						Ú					
Icelandic					€								µ				@									
Italian					€																					
Latvian	Ā		Č		Ē		Ģ		Ī		Ķ	Ļ		Ņ	Õ			Ŗ	Š		Ū					Ž
Lithuanian					€																					
Lithuanian (Standard)					€	“											|						}	%		
Maltese	À				È				Ì						Ò						Ù					
Norwegian					€								µ													
Polish		{		Đ									§	}			\		đ		€	@	¦			
Portuguese					€																					
Romanian		{		Đ							ł	Ł	§	}			\		đ			@	|			
Romanian (Standard)				Đ	€							Ł				§			ß							
Slovak		{	&	Đ	€	[	]				ł	Ł		}		‘	\		đ			@	|	#		>
Slovenian		{			€	[	]				ł	Ł	§	}			\					@	|			
Spanish					€																					
Swedish					€								µ													
Turkish	Æ				€				İ								@		ß							

Table 6: The characters on the A…Z keys that are typed using the third level of keyboard in different languages

XVZ CAOSW NPFQBIU YGJLRETHDKM.
Letters are split into four groups according to the '1' marks in Table 5:

1st group: marked with 1 in row 2;

2nd group: marked with 1 in rows 3 and 4;

3rd group: marked with 1 in row 3 only;

4th group: not marked 1 in any row.

4. Special Characters on the Third Level of Keyboard Layouts

According to the international standard ISO/IEC 9995, the third level of the keyboard is used to type characters that are used quite rarely. In all of the European languages that use the Latin script, with the exception of US English, three keyboard levels are used. To type a character, assigned to the third level, the key of that character should be pressed while holding the third level key (*Alt Gr*). In previous keyboards without the third level keys the key could be simulated by the key combination *Ctrl+Alt (Left)*. Such a key combination together with a character could be potentially used for command keys. This causes no difficulties if there are no such characters assigned to the third level of a particular language keyboard. But if a character is assigned to the third level, the action of typing that character will be blocked by the command key combination. Therefore, it is important to find out which keys have third level characters. The results of the analysis of various language keyboard layouts are presented in Table 6.

Table 6 clearly indicates that only three keys (*H, T*, and *Y*) do not have third level characters in all keyboards of the languages analysed. Therefore, the key combination *Ctrl+Alt* will not cause any conflict for these letters if used in the command keys. The third level of all other 23 letters is occupied in at least one language keyboard; therefore, the combination *Ctrl+Alt* should not be used with these letters. It is likely that the fact that the three letters mentioned are not occupied is just a coincidence. Therefore, in general, the pair *Ctrl+Alt* should not be used at all for any keys. If there is at least one key combination of *Ctrl+Alt* and a letter, this should be considered as an internationalisation error. Such combinations will not raise any problems in localised programs if the keyboard of the localisation target language does not have any third level characters on the key marked with that letter.

The keyboards of almost all languages have third level characters on all numeric keys. Therefore, the *Ctrl+Alt* combination used with any numeric character for the command key should also be treated as an internationalisation error. Thus we arrive to more radical solution: suggesting that the replication function from *Ctrl+Alt* to *Alt Gr* be removed from keyboard driver generating tools.

5. Key Combinations Using the Second Level Key

Letter keys in shortcut keys are usually indicated as capital letters, however they are typed as small letters, i.e. without the second level key (e.g., *Shift*). This does not cause any misunderstandings as letters on the keys are also presented as capital letters. In fact, the letters are used as key names but not as letters. Thus, the command keys are not influenced by the "caps lock" state.

Sometimes the second level key is used in command key combinations. In such a case, the second level key does not change the level of the keyboard, and it is included in the key combination in an expressed way, e.g. it is shown as *Ctrl+Shift+A* and explained as: "press control key, shift key, and letter A key", rather than "press control key and the capital letter A".

The second level key is rarely used together with letter keys in command key combinations and does not form a strong relationship as discussed above; therefore, such command keys may be localised freely. Sometimes, the second level key is used in the localised version of command keys so that the localised key does not cause any conflict with other, non localised keys. An example of such a case would be the usage of the combination *Strg+Umschalt+F (Ctrl+Shift+ F)* in German localised software. The combination is applied to invoke the command that changes the font of the selected text to bold (*Fet* in German). Such a key is selected in order to retain the letter *F* in a non-localised key combination for *Search* command *Strg+F* (*Ctrl+F*). It is less comfortable to press two keys (*Ctrl+Shift*) instead of one but it is likely that it is viewed as being easier than using a non-localised key, otherwise localisation would have been in vain.

Numeric row keys are also named by numbers, regardless of which level of the keyboard is used to type them. In the English language keyboard numbers are presented on the first level. The same is applicable for keyboards in the majority of other

languages. Therefore, they should not be localised because the number mnemonics, if used, are similar or the same in many languages. For example, *Ctrl+5* can mean a browser command *Open Tab 5*. However in languages where numbers are presented on the second level of the keyboard (e.g. French, Lithuanian), a localiser will need to modify the program so that the key combination *Ctrl+n* would have the meaning of pressing a control key and the key of number *n* (i.e. so that the second level key does not have to be pressed).

6. Command Key Characters in Keyboard Drivers

In keyboard drivers, characters that are used in combinations of command keys are defined separately from the characters that are typed by pressing keys to produce a regular text. Therefore, a keyboard driver can be developed to distinguish the layout of command keys and the keyboard. For example, the QWERTZ keyboard is used in Germany. In this layout of the keyboard, the letter *Z* is on the same key as the letter *Y* in the layout of the QWERTY keyboard. Therefore exhausting the possibilities of the development of the keyboard driver discussed above, the QWERTZ keyboard can be developed for Germany as follows: The command *Undo*, marked by *Ctrl+Z* could be called by pressing either the letter *Z*, to conform to the German keyboard, or the letter *Y*, to conform to the position of the letter Z in the English keyboard. Moreover, any other letter key that corresponds to the position of the letter *Z* could be used as well. Let us look at how this is carried out in practice.

During our research, the layouts of the German QWERTZ and the French AZERTY keyboards were tested and it was noted that neither of them distinguish between the position of the keys that are used for command keys, and those that are used to type regular text. This decision is natural. The "unnatural" layout (as in the example above) is required in special cases, for example, with languages that use a non-Latin script but apply Latin letters in command keys when the keyboard is phonetic and is used in other language environments.

7. Conclusions

Our investigation shows that the bindings of the command keys to letters in original programs are rather stable, i.e. the command is related to the same letter in all programs (with some rare exceptions). It is advisable to maintain such a positive feature in localised programs, implementing a coordinated adjustment of letters in all localised programs for a particular language.

One letter is usually related to several commands since there are more commands than letters in the alphabet. However, in general there are dominant bindings of letter and command. According to the findings of our analysis of 50 frequently used programs, a list of dominant command-letter pairs has been developed. If a command letter, included in this list, is localised, the coordination of its adjustment in all other localised software is more important and more difficult in comparison to the localisation of other letters.

According to localisation requirements and possibilities the command keys may be categorized into three groups:

1) Letters that do not need to be localised;
2) Group of letters to be used on the basis of national level agreements (23 commands make up dominant pairs of command letters);
3) A group of letters that have to be used according to local agreements (119 commands that are not included in the list of dominant commands).

It has been observed that almost every character key, on keyboards based on the Latin alphabet, is used to type third level characters. Since the key combination *Ctrl+Alt (left)* simulates the third level key (*Alt Gr*), the usage of this combination for command keys must be considered as an internationalisation defect. This defect would lead to a block on typing characters that are assigned to the third level of the keyboard.

Acknowledgements

This work has been partly supported by the Lithuanian Science Council Student Research Fellowship Award (Agnė Strelkauskytė).

References

Esselink, B. (2000) *A practical guide to localization.* Amsterdam: John Benjamins.

Grigas, G. and Strelkauskytė, A. (2011) 'Localisation problems of shortcut keys (Sparčiųjų

klavišų lokalizavimo problemos)'. Proceedings of XV conference of computer scientists. Klaipėda, Lithuania, September 22–24, 2011. Vilnius: Žara, 64-75 [in Lithuanian].

Hall, V. and Hudson, R. (1997) *Software without frontiers.* Willey & Sons.

Keyboard Shortcut (2012) *Wikipedia, the free encyclopedia* (online), available: http://en.wikipedia.org/wiki/Keyboard_shortcut [accessed 24 October 2012].

Microsoft Corporation (2010) Microsoft Language Portal (online), available: http://www.microsoft.com/Language/en-US/Default.aspx [accessed 24 October 2012].

Microsoft Corporation (2007) *Microsoft Office Word Help*. Keyboard shortcuts for Microsoft Office Word.

Müler, E. (2009) 'Building Quality into the Localization Process'. *Multilingual*, May-April, 14-15.

Safar, L. and Machala, J. (2010) 'Best practices in localization testing'. *Multilingual*, January-February, 1-4.

Schäler, R. (2003) 'The Cultural Dimension in Software localisation'. *Localisation Reader 2003-2004. Selected articles from Localisation Focus and Multilingual Computing &Technology* (online), 5–8, available: http://www.localisation.ie/resources/Research/ELECT/Consortium/ContentFiles/-00-11%20LR-S.pdf [accessed 24 October 2012].

Table of keyboard shortcuts (2012). *Wikipedia, the free encyclopedia* (online), available: http://en.wikipedia.org/wiki/Table_of_keyboard_shortcuts [accessed 24 October 2012].

A Flexible Decision Tool for Implementing Post-editing Guidelines

Celia Rico Pérez
Departamento de Periodismo y Comunicación Intercultural
Facultad de Arte y Comunicación
Universidad Europea de Madrid
Madrid, Spain
www.uem.es
celia.rico@uem.es

Abstract

This paper presents a flexible tool which supports the decision making process involved in defining post-editing guidelines. It is inspired by the work of O'Brien (2012) towards a translation quality assessment model, briefly reviewed here with a view to adapting it to post-editing specifications. The paper first presents a review of the relevant literature on post-editing, followed by an account of the challenges involved in defining post-editing rules. The different components of the decision tool are then presented and illustrated with examples. Finally, a discussion of its effectiveness is advanced together with some indications for further work.

Keywords: *post-editing, machine translation, guidelines for post-editors*

1. Introduction

One of the crucial aspects in a Post-editing (PE) project is to decide on guidelines to be followed by post-editors. Selecting what elements to change and delivering a final text at a sufficient level of quality is usually a matter of difficulty, due to the subjectivity involved in the task and closely related to specifying a desired quality level. Acceptance and use of half –or semi-finished– texts determine to what extent MT output should be post-edited, and how much human effort is necessary to improve such imperfect texts (Allen 2003, p. 301). In this respect, MT acceptability and its correlation to human effort has been the subject of considerable discussion in the relevant literature (Fiederer and O'Brien 2009, Guerberof 2009, Roturier 2004), reporting significant findings both in automatic metrics (Quirk 2004, Specia 2011, Specia et al 2009, Specia and Farzindar 2010, among others) and human assessment (as, for instance, in Garcia 2011, O'Brien 2011b, Thicke 2011).

Specifying PE guidelines involves, then, deciding on text quality acceptance which, in turn, depends on aspects such as client expectations, turn-around time or document life-cycle, among others. Usually, approaches to PE take as a point of departure the distinction between full and light PE (Allen 2003, TAUS/CNGL 2011), with various levels of PE being implemented in different settings and contexts: for research purposes (de Almeida and O'Brien 2010, Garcia 2011, O'Brien 2011b, Roturier 2004, Specia and Farzindar 2010, to name but a few); in audiovisual translation (de Sousa et al 2011); for reviewing official languages translation of government documents and institutional translation (Aymerich and Camelo 2006, Bowker 2009); and in commercial settings (Beaton and Contreras 2010, Plitt 2011), among others.

Nevertheless, this division between full and light PE might get somewhat blurred as human post-editors generally tend to engage in full post-editing (O'Brien 2011a), deeming this dissociation as irrelevant. In this context, and with the observation that Machine Translation is compelling the translation industry to search for new business models, it seems appropriate to explore new approaches to PE. With MT engines leaving the research labs and opening up to broader and generalized practice –contrasting with previous implementations in highly specialized technical contexts– MT is now a real alternative to human translation even in commercial contexts where it was not used just a decade ago. Reports indicate a substantial increase in the use of MT among Language Service Providers of which "41% of Best-in-class use Machine Translation as a component of their translation process" (Houlian 2009, p. 12), resulting in an average cut by 15% in their translation

time. In this context, when MT is broadly used for almost any purpose it is only natural that "post-editing strategies have to evolve in line with the technology" (Beaton and Contreras 2010).

This paper presents a decision tool for designing PE guidelines in an attempt to offer a flexible framework for arriving at informed decisions and considering all relevant aspects in a PE project. The focus is placed on the practicalities of implementing such a tool in real scenarios. In the sections that follow, I review first the main challenges to be addressed when defining PE guidelines, and introduce some of the recommendations usually followed. Then, the decision tool is explained with actual examples from its empirical use in the context of the research project EDI-TA.

2. The challenge in defining PE guidelines

When it comes to establishing PE guidelines to be implemented in a real-world scenario with a decisive impact on costs, turnaround time and quality, directions provided by the relevant literature on the subject seem somewhat sparse. PE specifications are either general recommendations that need further development, or rules specifically tailored for a particular PE project, which cannot be replicated, in a different scenario, without difficulty.

In drawing PE guidelines, the typical approach, then, is to proceed considering a series of aspects such as client, volume of documentation to be processed, quality expectation, turnaround time, document life expectancy, and use of the final text (Allen 2003, p. 301; O'Brien 2011a, p. 4). From then on, a distinction is made in *rapid*, *partial* or *full* post-editing, with expectations on translation use playing a key role in the definition of correction strategies. Hence, an *inbound translation approach* would lead to either MT with no post-editing (when texts are used for information browsing) or rapid PE (for perishable texts). On the other hand, an *outbound translation approach* would compel partial or even full PE, depending on the quality of the translated output and the final use of the text.

Actual implementations of these principles, both in the translation industry and in experimental settings for research purposes, range from early PE methodologies, put into practice about a decade ago, to more recent initiatives responding to the late growth in the use of MT. Examples of the former are the use of error correction guidelines which take as a model standards such as SAEJ2450 at General Motors, MT system specific guidelines used at the Pan-American Health Organization, or rules specific to the European Commission Translation Services (PE case-studies as reported in Allen 2003). More recently, PE initiatives include the integration of machine translation with a commercially available post-editing solution, (Beaton and Contreras 2010), measuring PE effort related to MT output quality (Guerberof 2009, Thicke 2011, Plitt and Masselot 2010, Roturier 2004, Specia and Farzindar 2010, Specia 2011), automated post-editing (Lawson-Tancred 2008), assessing and developing PE tools (Vieira and Specia 2011, Aziz et al 2012), post-editing as a viable alternative to conventional translation (Garcia 2011), and estimating productivity (Guerberof 2008, O'Brien 2011b). These authors mention guidelines scarcely and, in most cases, these are usually taken for granted. This situation reveals that internationally adopted standard guidelines are still to be defined as each company tends to have their own PE directions (O'Brien et al 2009).

In this context, it still holds true that post-editors need specific linguistic and technical directions that help them overcome uncertainty and take the appropriate decision when confronted by the task with a "certain degree of tolerance and the ability to draw clear boundaries between purely stylistic improvements and required linguistic corrections" (Krings and Koby 2001, p. 16). After all, "what most people really want to know is what are the actual post-editing principles that support the post-editing concept" (Allen 2003, p. 306). It is true that many important advances have been made in the field with numerous experiments and case-studies being conducted, yet my contention is that a flexible decision tool is still needed, one that considers text characteristics, language specific rules and system specific recommendations.

3. A flexible decision tool

Establishing clear guidelines for a PE project involves the consideration of the following aspects (Guerberof 2010, p. 35): type of MT engine, description of source text, reference to output quality and client's expectations, scenarios indicating when to discard a segment, typical errors to be corrected, changes to be avoided, and specifications on how to deal with terminology. Additionally, there is one recommendation which holds true in any PE project, and that is "specifying the scope of manual MT post-editing and sticking to it stoically". Otherwise "vast amounts of time, effort and money are unnecessarily

spent in making merely stylistic corrections" (Guzmán 2007). In this sense, Guzmán advances a series of practical aspects to be considered when training post-editors, two of which refer specifically to the design of PE guidelines, namely:

1 Create clear guidelines with detailed examples of what needs and does not need to be post-edited.

2 Anticipate potential issues and appropriate solutions, with an emphasis on how to deal with stylistic and terminology inconsistencies.

However, before one can really stick *stoically* to a series of specifications, thorough consideration is needed on all the aspects involved, if possible adhering to a comprehensive model that serves as a decision tool. One such tool is advanced here, adapting the *dynamic quality evaluation model for translation*, as devised by O'Brien (2012) in her benchmarking exercise carried out in collaboration with the Translation Automation User Society (TAUS).

In this exercise O'Brien first reviews error detection categories and, after comparing eight quality evaluation models actually used in the translation industry, concludes that there is a significant level of agreement in the macro and micro categories for error detection but that "penalties and weightings applied differed from one model to the other […] with a preference for a segment-level error analysis over a holistic user-focused evaluation" (p. 64) which overlooks relevant characteristics such as text type, user requirements or perishability. The study goes on to review how quality is measured in the areas of Machine Translation, Translator Training, Community Translation and Technical Communication. The evaluation models identified are as follows: adherence to regulatory instruments, usability evaluation, error typology, adequacy/fluency, community-based evaluation, readability evaluation, content sentiment rating, customer feedback. The proposal for a dynamic model is based, then, on two building blocks: *communication channel* and *content profile.* The first distinguishes two possible channels of communication: one where information is used for internal purposes and the other where information is conceived for external use. This latter channel is subdivided into three other channels: Business to Consumer (B2C), Business to Business (B2B) and Consumer to Consumer (C2C). This distinction helps determine quality expectations from the client as, for instance, a document to be consumed internally might call for lower quality expectations than one devised for external communication.

With regards to the second building block, content profile, the model identifies eight meta-categories for content type: user interface text, marketing material, user documentation, website content, online help, audio/video content, social media content, and training material. Each of these types is then mapped onto the parameters of utility, time and sentiment (UTS ratings), defined as follows: "*utility* refers to the relative importance of the functionality of the translated content; *time* refers to the speed with which the translation is required; and *sentiment* refers to the importance of impact on brand image" (p.71).

O'Brien's dynamic model proceeds to assign "the person in charge of the quality evaluation" with the task of identifying the communication channel and content profile, together with the responsibility for rating the text in terms of utility, time and sentiment. Once this is done, the next step is to consider what is involved in each QE model and decide, on the basis of contextual factors, which model to apply" (p.72). By way of an example, we learn that training material, which is classified as Internal communication channel, is tagged medium for Utility, high for Time, and low for Sentiment, which leads to two recommended QE models in descending order of control: 1) adequacy/fluency, 2) (internal) community-based evaluation. These are then mapped onto evaluation parameters resulting in a proposal for a "more dynamic QE model" (p.72) with "an impact on quality expectations".

Upon review of this model for translation quality assessment, I found out that contextual aspects as defined by O'Brien, affect PE projects similarly and that those drawbacks she identifies in static models also hold true to PE estimation, where "current models are predicated on a static and serial model of translation production […] with little consideration given to variables such as content type, communicative function, end user requirements, context, perishability, or mode of translation creation" (p.55).

The remainder of this paper presents a model that fosters the development of these kinds of PE specifications. Designed as a decision tool that would support post-editors in defining PE rules, the model will aid post-editors in their constant "struggle with the issue of the quantity of elements to change while also keeping the translated text at a sufficient level of

quality" (Allen 2003). In this respect, the tool aims at gathering, in a single source, all aspects influencing the post-editor's decision so that PE guidelines can be easily drawn, adequately supported with actual examples and, more importantly, shared and replicated along different PE projects.

The main elements of the decision tool are listed in figure 1 for the sake of clarity, with a thorough explanation and examples in the subsequent sections.

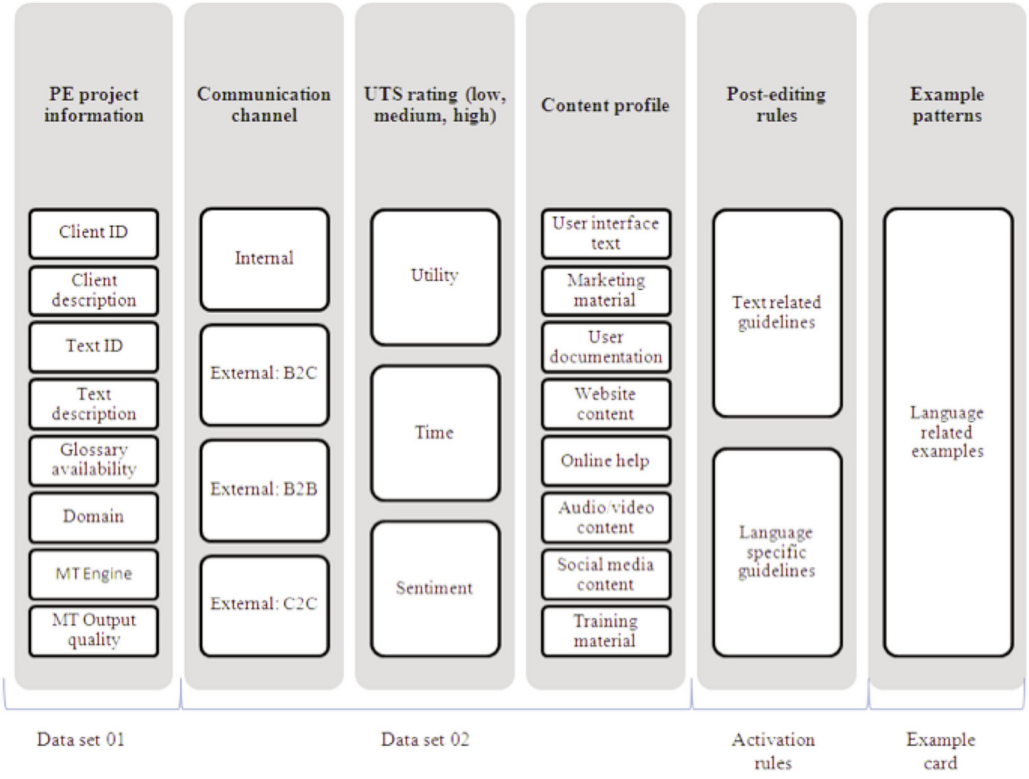

Figure 1: Elements in the PE decision tool

These building blocks are the basis of the PE decision tool but instead of mapping them to evaluation parameters, correspondences are drawn between them and PE rules, as illustrated below. These elements are grouped into two data sets (PE project information and text profile), two rule activation sets (text related guidelines and language specific rules) and an example card for registering typical PE samples. The data sets provide practical information on the PE project as well as a formalization of other aspects which, subsequently, contribute to specifying PE guidelines. Finally, the example card provides actual examples of how each rule is to be applied. The sections that follow offer a detailed explanation of the decision tool.

3.1 Data set 01: PE project information

This data set collects information on the PE project and allows the project coordinator to keep track of its most practical aspects, while gathering both evaluative and descriptive knowledge on the task at hand. The list of features to be considered is defined as follows:

- Client identification (this refers to the internal project identification code)
- Client description (this is a short description of who the client is together with any particularities the project coordinator might deem necessary)
- Text identification (this would typically be an internal project code)
- Text description (a short description of the particularities of the text not covered in any of the other categories)
- Glossary availability (indicating whether there are any available glossaries –from the client or internal to the company- which the

post-editor might need access to; and information on glossaries completeness and quality)

- Domain (this refers to the specification of content subject area)
- MT engine (this is a reference to the MT system used, with indication of any specific internal rules which have been activated, glossaries used, training data, and interaction with translation memories, if any)
- MT output quality (this refers to a grading of the output text quality)

For each of the categories in this data set, the project coordinator should indicate the appropriate information as illustrated in Table 1. This will be used later for defining rules to be activated and assessing PE effort with a view to planning work.

As we can see, this set of data gathers information on project particularities which may either affect the decision process or be pertinent for keeping track of the decision, should post-editors need further clarification about their task. For instance, information related to client description is useful for the post-editor to gather knowledge on the project. Similarly, if glossaries are available they would need to be considered right from the beginning so that post-editors can actually use them. This would help them in checking the appropriate terminology when needed and/or adding new terms (should this task be assigned to them). In addition, information related to the MT Engine is relevant for experienced post-editors as it gives them an insight on what to expect from the output. In this respect, metrics to be applied to the category "MT output quality" are either

PE project information	
Client ID	*<indicate client identification according to the company's own codification system>*
Client description	*<provide a short description of the client>*
Text ID	*<indicate text identification according to the company's own codification system>*
Text description	*<provide a short description of the text contents>*
Glossary availability	*<indicate the availability of glossaries associated to the PE project> <indicate whether glossaries are completed or need revision/updating> <indicate whether glossaries have been quality controlled>*
Domain	*<indicate text domain>*
MT Engine	*<indicate the MT engine used> <for rule-based MT systems, indicate any specific rule that has been activated> <indicate if domain/client glossaries have been activated > <for statistical MT engines, indicate set of training data used> <indicate type of interaction (if any) with translation memories> <indicate how domain-specific untranslatable entities have been handled>*
MT Output quality	*<grade MT output according to quality metrics>*

Table 1: Data set 01: PE project information

automatic metrics such as BLEU (Papineni et al 2002) or METEOR (Banerjee and Lavie 2005) or those following human judgment (O'Brien 2012, Garcia 2011, Roturier 2004, Estrella et al 2007). What is relevant here is that the MT output quality holds a direct relationship with the expected PE effort (Specia 2011) and that information provided in this data set provides a context on how the automatic translation has been approached, thus influencing the later choice of PE rules.

3.2 Data set 02: Text profile

Text profile is defined on three main categories, based on O'Brien's (2012) dynamic model for quality assessment:

- Communication channel. This refers to the description of the communicative purposes of the document, which can be used either for internal purposes or for external communication, as previously described. This latter category is further divided into three subcategories: Business to Customer, Business to Business and Customer to Customer.
- Content profile. The information gathered in this category relates to text type and complements that of data set 01 regarding "text description" and "domain".
- Utility, Time and Sentiment. These subcategories refer to the importance of the functionality of the translated content (Utility), the speed at which the PE output is to be handled (Time), and the importance of impact on brand image. Each of these are rated according to three metrics: low, medium and high.

For each of the categories in this data set, the project coordinator should indicate the appropriate information, as illustrated in Table 2 so that, later, PE rules can be correspondingly activated.

3.3 Rule activation set 01:Text related guidelines

The rule set shown in table 3 is an attempt at formalizing typical errors to be corrected in the MT output. These are taken from general guidelines as depicted in O'Brien (2011a) and more specific ones, as mentioned in Guzmán (2007), and Torrejón and Rico (2002). The aim is to offer clear indications to post-editors on how to proceed when confronted with text to be post-edited.

The way to proceed is to review each of the rules and decide whether to activate them or not, depending on the information previously gathered in the two data sets above.

Text profile	
Communication channel	*<indicate the text communication channel: Internal (I); External: Business to Customer (B2C); External: Business to Business (B2B); External: Customer to Customer (C2C)>*
Content profile	*<indicate content profile from the following list: User interface text, Marketing material, User documentation, Website content, Online help, Audio/video content, Social media content, Training material>*
UTS rating (low, medium, high)	*<indicate rating for Utility> <indicate rating for Time> <indicate rating for Sentiment>*

Table 2: Data set 02: Text profile

3.4 Rule activation set 02:Language specific guidelines

Together with the general PE guidelines activated in the data set above, there might be some language specific guidelines (table 4) that need to be taken into consideration, when they are not covered in text related guidelines.

Text related guidelines	
Fix wrong terminology	*<indicate whether this rule should be activated>*
Spend time in terminology research	*<indicate whether this rule should be activated>*
Fix syntactic errors (wrong part of speech, incorrect phrase structure, wrong linear order)	*<indicate whether this rule should be activated>*
Fix morphological errors (number, gender, case, tense, voice)	*<indicate whether this rule should be activated>*
Fix misspelling errors	*<indicate whether this rule should be activated>*
Fix punctuation errors	*<indicate whether this rule should be activated>*
Fix any omissions as long as they interfere with the message transferred	*<indicate whether this rule should be activated>*
Edit any offensive, inappropriate or culturally unacceptable information	*<indicate whether this rule should be activated>*
Fix any problem related to textual standards (cohesion, coherence)	*<indicate whether this rule should be activated>*
Make explicit any necessary information	*<indicate whether this rule should be activated, if the source text needs to be clarified or made more explicit in the target text>>*
Fix stylistic problems	*<indicate whether this rule should be activated>*

Table 3: Rule activation set 01: Text related guidelines

Language specific guidelines
<indicate any language specific guidelines to be taken into account>

Table 4: Rule activation set 02: Language specific guidelines

Language specific rules are, for example, the use of a particular language locale, lexical collocations or specific sentence structures, how product names should be dealt with (whether there is an equivalent available or the source language name should be used). In the language combination ES-EN, rules would typically include instructions on how to deal with the translation of sentences using the infinitive tense, how to PE third person singular, or an indication of when to delete unnecessary uses of "the", among others.

3.5 Example card

As already mentioned, it is key to provide post-editors with a set of representative examples for each of the rules so that they know what to look for, how to deal with the different rules and what PE implies. Some examples are provided (tables 5 to 9) by way

of illustration on how to use the example card in the language pair Spanish-English. Each PE project should compile its own example card, taking into account the particularities of the text, language combination, as well as all other parameters as we have seen above. Each language pair should provide its own examples. Other interesting examples can found in Guzmán (2007), Guerberof (2008) and Thicke (2011), among others.

PE rule	*MT input*	*MT output*	*PE output*
Fix wrong terminology	• Sin derecho a deducción	• Without **law** to deduction	• Without the **right** of deduction
	• Place the fuel filler cap in the bracket, which is attached to the fuel filler door.	• Coloque la **gorra** de relleno de combustible en el soporte atado a la puerta de relleno de combustible	• Coloque la **tapa** del depósito de combustible en el soporte que hay en la puerta del depósito.

Table 5: examples illustrating how to fix wrong terminology (language pair ES-EN)

PE rule	*MT input*	*MT output*	*PE output*
Fix syntactic errors (wrong part of speech, incorrect phrase structure, wrong linear order)	• Para aprender más cosas sobre la ciencia	• To **learn** more things **on** the science	• To learn more things **about** science
	• Planes de prevision asegurados	• Plans of **forecast guaranteed**	• Plans of **guaranteed forecasts**
	• Place the fuel filler cap in the bracket, which is attached to the fuel filler door.	• Coloque la gorra de relleno de combustible en el soporte **atado** a la puerta de relleno de combustible	• Coloque la tapa del depósito de combustible en el soporte **que hay en** la puerta del depósito

Table 6: examples illustrating how to fix syntactic errors (language pair ES-EN)

PE rule	*MT input*	*MT output*	*PE output*
Fix morphological errors (number, gender, case, tense, voice)	• You can connect the audio devices to the USB audio interface	• Usted puede conectar dispositivos de audio **con el** interfaz de audio USB	• Usted puede conectar dispositivos de audio **a la** interfaz de audio USB

Table 7: examples illustrating how to fix morphological errors (language pair ES-EN)

PE rule	*MT input*	*MT output*	*PE output*
Fix any omissions as long as they interfere with the message transferred	• The containers could become leaky and cause an explosion or a fire	• Éstos [] podrían hacerse agujereados y causa y explosión	• **Los contenedores** podrían tener filtraciones y causar una explosión.

Table 8: examples illustrating how to fix omissions (language pair ES-EN)

PE rule	*MT input*	*MT output*	*PE output*
Fix stylistic problems	• The USB audio interface	• Interfaz de audio **de** USB	• La interfaz de audio USB

Table 9: examples illustrating how to fix stylistic problems (language pair ES-EN)

3.6 A brief note on the tool implementation

The implementation of this tool rests on the assumption that there is a project coordinator who takes responsibility for filling in the data sets, activating the rules and providing adequate examples. The process would typically start by indicating relevant information in data sets 01 (PE project information) and 02 (Text profile), which is the basis for deciding on rules to be activated (both text related and language specific ones) and, finally, illustrating them with examples. Together with these, training should be conducted to anticipate potential problems and appropriate solutions such as how to proceed with stylistic inconsistencies, terminology misuse or deciding what linguistic patterns would require severe post-editing.

Additionally, it is recommended that a PE kit should be prepared with the following materials:

- Project information (as contained in data sets 01 and 02)
- Specific guidelines on how to proceed, indicating what rules should be activated
- Specific examples detailing each of the rules
- Access to the project's glossaries
- A reporting card for any feedback post editors might find useful for subsequent PE projects (language specific rules not originally

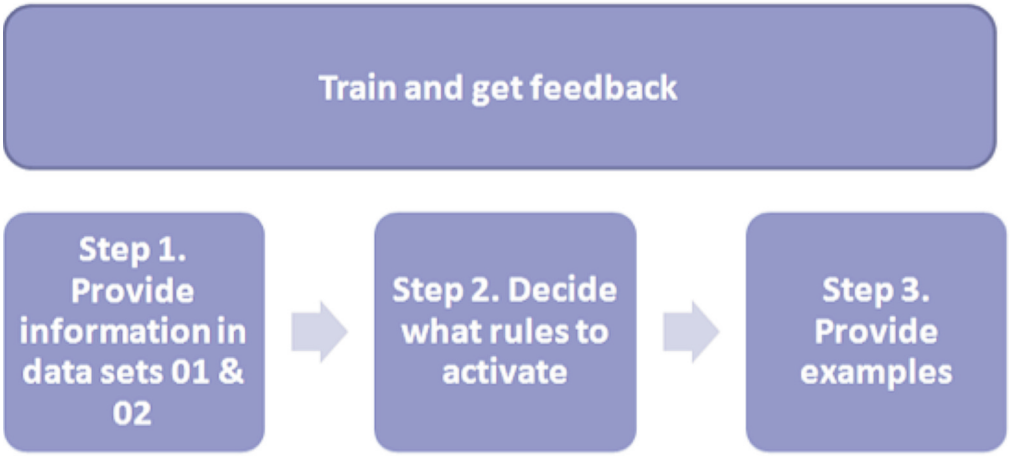

Figure 2: steps in the tool implementation

contemplated, linguistic structures and terms to be reported for improvement of the MT engine).

- The MT output
- The MT input

The whole process is summarized in figure 2.

4. Discussion and further work

The tool presented here originates from work carried out in the context of EDI-TA, a research project set out with the following objectives:

1 Defining metadata suitable for post-editing purposes
2 Testing the contribution of metadata for improving post-editing processes
3 Defining a practical methodology for post-editing between distant languages pairs, namely, Spanish into English, French, German and Basque, and from English into Spanish.
4 Suggesting improvements in the MT system so as to optimize the output for post-editing specific purposes
5 Showing the feasibility and cost reduction of implementing post-editing in a real scenario
6 Identifying functions for improving post-editing tools
7 Define a methodology for training post-editors in the project's language pairs (namely ES, EN, FR, EU, and DE)

The full description of work carried out at EDI-TA is reported in Rico and Díez (2012) and a complete account of results can be consulted in Rico (forthcoming). Work towards the design, implementation and test of the PE decision tool, as described in the present paper, was addressed into two subsequent phases:

Phase 1. Post-editing pilot project start-up.

This phase focused on setting up a pilot test that would serve as a reference in subsequent phases of the project. Core tasks included:

a) Web content selection. A first set of web content was selected for this pilot test. Language pairs were EN-ES, EN-EU, EN-FR, ES-EN, and domains referred to online customer information in mobile technology, and information for citizens in the Spanish Internal Revenue Service.

b) Text analysis for post-editing. This involved the identification of different aspects that might involve some kind of problem for post-editing purposes as well as the registration of MT output errors.

The outcome of this first phase was a tentative set of PE guidelines whose effectiveness was to be tested in the next phase

Phase 2. MT post-editing experimentation.

This phase focused on conducting a PE experiment on the basis of the findings above. The following tasks were carried out:

a) Creating a PE project. This involved selecting a new set of web content. This time the domain referred to information from the Spanish public administration. Language combinations were as follows: EN-ES, EN-EU, EN-FR, ES-EN. This set amounted to a total of 50,000 words per language pair

b) Error analysis. MT output was evaluated so as to detect possible errors which affect PE (lexical, syntactic, terminological).

c) Definition of post-editing rules. PE guidelines were specified with the help of the dynamic model as mentioned above. These included explicit references on what to expect from the MT output in terms of quality and how to proceed in each case.

d) Testing PE guidelines effectiveness. A comprehensive list of PE specifications were put to test in the PE project with the language combinations and domain as described above.

As a result of this experiment, a guide containing practical information on how to approach a PE project was designed. The tool is conceived as a flexible framework to cater for different PE projects and scenarios. In this respect, lack of information and documented procedures on how PE should be carried out is still reported among industry players (Lucardi 2012) where guidelines tend to be either too general or too context-specific to be replicated straightaway.

In this experiment, findings reported that further refinement and training of the model is still needed so

as to overcome the subjective point of view of the person who uses it. Even so, it provides a methodology for guidelines specification which can be formally shared, and includes examples and clear guidance on how to proceed.

In this sense, the tool is a valuable instrument as it collects, in a single source, all aspects influencing the post-editor's decision so that PE guidelines can be easily drawn up, adequately supported with actual examples and, more importantly, shared and replicated along different PE projects. Other rules might be added, particularly with reference to language guidelines, and further research is also needed towards defining the PE kit and evaluating its efficiency in different settings.

Acknowledgements

The work presented here has been carried out in the framework of EDI-TA, a project funded by Linguaserve R&D programme, as part of the tasks it is developing within The MultilingualWeb-LT (Language Technologies) Working Group (http://www.w3.org/International/multilingualweb/lt/), which belongs to the W3C Internationalization Activity and the MultilingualWeb community. The MultilingualWeb-LT Working Group receives funding by the European Commission (project name LT-Web) through the Seventh Framework Programme (FP7) Grant Agreement No. 287815.

I am gratefully indebted to the members of the EDI-TA team for their valuable insights as to how to implement this model in a real context, participating in fruitful discussions and contributing with relevant examples. The team is made up of the following members: Lidia Cámara, Igone Regidor, Johanna Blasco, Martín Ariano and Félix Fernández. I also take this opportunity to acknowledge the strategic vision of Pedro L. Díez-Orzas, Linguaserve's CEO, in setting up this project.

References

Allen, J. (2003) 'Post-editing' in Somers, H. ed., *Computers and Translation. A translator's guide*, Amsterdam/Philadelphia: John Benjamins, 297-317.

Aymerich, J. and Camelo, H. (2006) "Post-Editing of MT Output in a Production Setting: Experiences at the Pan American Health Organization", *AMTA 2006* [online], Cambridge, MA. August, available: http://www.paho.org/english/am/gsp/tr/mt_docs.htm [accessed 2 November 2012]

Aziz, W., de Sousa, S. C. M. and Specia, L. (2012) 'PET: a tool for post-editing and assessing machine translation', paper presented at *The Eighth International Conference on Language Resources and Evaluation,* LREC '12, Istanbul, Turkey, May 2012

Banerjee, S. and Lavie, A. (2005) 'METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments´', in *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43rd Annual Meeting of the Association of Computational Linguistics (ACL-2005),* Ann Arbor, Michigan, June 2005

Beaton, A. and Contreras, G. (2010) 'Sharing the Continental Airlines and SDL post-editing experience', paper presented at *AMTA 2010, The Ninth Conference of the Association for Machine Translation in the Americas* [online], Denver, Colorado, October 31 – November 4, available: http://www.mt-archive.info/AMTA-2010-Beaton.pdf [accessed 28 June 2012]

Bowker, L. (2009) 'Can Machine Translation Meet the Needs of Official Language Minority Communities in Canada? A Recipient Evaluation' *Linguistica Antverpiensia* 8, 123-155.

de Almeida, G. and O'Brien, S. (2010) 'Analysing Post-Editing Performance: Correlations with Years of Translation Experience', *EAMT May 2010 St Raphael, France* [online], available: http://www.mt-archive.info/EAMT-2010-Almeida.pdf [accessed 2 November 2012]

de Sousa, S.C.M., Aziz, W. and Specia, L. (2011) 'Assessing the post-editing effort for automatic and semi-automatic translations of DVD subtitles', *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011* [online], available: http://clg.wlv.ac.uk/papers/ranlp-2011_sousa.pdf [accessed 2 November 2012]

Estrella, P., Popescu-Belis, A. and King M. (2007) 'A New Method for the Study of Correlations between MT Evaluation Metrics' in *Proceedings of TMI-07 (11th Conference on Theoretical and Methodological Issues in Machine Translation*), Skövde, Sweden.

Fiederer, R. and O'Brien, S. (2009) 'Quality and Machine Translation: A realistic objective?', *The Journal of Specialised Translation* [online], issue

11, January, 52-74, available: http://www.jostrans.org/issue11/art_fiederer_obrien.pdf [accessed 28 June 2012]

Garcia, I. (2011) 'Translating by post-editing: is it the way forward?', *Machine Translation*, *25*(3), 217-237
Guerberof, A. (2008) 'Post-editing MT and TM. A Spanish case', *Multilingual*, 19 (6), 45-50.

Guerberof, A. (2009) 'Productivity and quality in MT post-editing', paper presented at the *MT Summit 2009 Workshop 3* [online], available: http://www.mt-archive.info/MTS-2009-TOC.htm [accessed 28 June 2012]

Guerberof, A. (2010) 'Project management and machine translation' *Multilingual*, 7 (1),34-38.

Guzmán, R. (2007) 'Manual MT Post-editing: if it's not broken, don't fix it', *Translation Journal*, (11), 4. October 2007.

Houlihan, D. (2009) *Translating Product Documentation: The Right Balance between Cost and Quality in the Localization Chain* [online], available: http://www.ptc.com/WCMS/files/108879/en/6230-RA-ManagingTranslation-DH-08-NSP.pdf [accessed 28 June 2012]

Krings, H.P and Koby, G.S. (2001) *Repairing Texts. Empirical Investigations of Machine Translation Pos-Editing Processes*. Kent: The Kent State University Press.

Lawson-Tancred, H. (2008) 'Monolingual translation: automated post-editing', *Multilingual*, April/May: 40-46.

Lucardi, G. (2012) 'Moses: The Trusted Translations Experience', *Taus Open Source Machine Translation Showcase* [online], available: http://www.slideshare.net/TAUS/4-june-2012-taus-moses-open-source-mt-showcase-paris-gustavo-lucardi-trusted-translations [accessed 28 June 2012]

O' Brien, S., Roturier, J., and Almeida, G. D. (2009) 'Post-Editing MT Output. Views for the researcher, trainer, publisher and practitioner', paper presented at the *MT Summit 2009 tutorial* [online], available: http://www.mt-archive.info/MTS-2009-OBrien-ppt.pdf [accessed 28 June 2012]

O' Brien, S. (2011a) *Introduction to Post-Editing: Who, What, How and Where to Next?* [online], available: http://www.cngl.ie/node/2542 [accessed 28 June 2012]

O'Brien, S. (2011b) 'Towards predicting post-editing productivity', *Machine Translation*, *25*(3), 197-215
O'Brien, S. (2012) 'Towards a Dynamic Quality Evaluation Model for Translation', *The Journal of Specialised Translation*, 17, Jan. 2012

Papineni, K., Roukos, S., Ward, T. and Zhu, W. J. (2002) 'BLEU: a method for automatic evaluation of machine translation", *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics* (ACL), Philadelphia, July 2012 311–318.

Plitt, M. and Masselot, M. (2010) 'A Productivity Test of Statistical Machine Translation' *The Prague Bulletin of Mathematical Linguistics*, 93, 7-16.

Plitt, M. (2011) *Post-editing tests at Autodesk* [online], available: http://stl.recherche.univ-lille3.fr/colloques/20112012/Mirko_Plitt_Lille_2012_02_03.pdf [accessed 30 October 2012]

Quirk, C. B. (2004) 'Training a Sentence-Level Machine Translation Confidence Measure', *Proceedings of the 4th International Conference on Language Resources and Evaluation*. Lisbon, Portugal 825-828.

Rico, C. (forthcoming) *EDI-TA: Post-editing Methodology for Machine Translation. Final report* [online] , available: http://www.w3.org/International/multilingualweb/lt/wiki/WP4#WP4:_Online_MT_Systems [accessed 30 October 2012]

Rico. C and Díez Orzas, P.L (2012) ?EDI-TA: Addressing Real Challenges in Post-editing?, *International Workshop on Expertise in Translation and Post-editing Research and Application* (ETP) [online], Copenhagen Business School, August 17 to August 18, 2012, available: http://www.cbs.dk/Forskning/Konferencer/ETP/Menu/Presentations [accessed 30 October 2012]

Roturier, J. (2004) 'Assessing a set of Controlled Language rules: Can they improve the performance of commercial Machine Translation systems?' Aslib ed., *Translating and the Computer 26*. London.

Specia, L., Cancedda, N., Dymetman, M., Turchi, M., and Cristianini, N. (2009) 'Estimating the Sentence-Level Quality of Machine Translation Systems', paper presented at the *13th Annual Meeting of the European Association for Machine Translation (EAMT-2009)*, Barcelona, Spain, 28-35.

Specia, L., and Farzindar, A. (2010) 'Estimating Machine Translation Post-Editing Effort with HTER', paper presented at the *AMTA-2010 Workshop Bringing MT to the User: MT Research and the Translation Industry* [online], Denver, Colorado, available: http://amta2010.amtaweb.org/ [accessed 28 June 2012]

Specia, L. (2011) 'Exploiting Objective Annotations for Measuring Translation Post-editing Effort', *Proceedings of the EAMT* [online], Leuven, Belgium, available: http://www.ccl.kuleuven.be/EAMT2011/ [accessed 28 June 2012]

TAUS/CNGL (2011) *Machine Translation Post-editing Guidelines* [online], available: http://www.translationautomation.com/machine-translation-post-editing-guidelines.html [accessed 28 June 2012]

Thicke, L. (2011) 'Improving MT results: a Study', *Multilingual*, (February), 37-40.

Torrejón Díaz, E. and Rico Pérez, C. (2002) 'Controlled Translation: A new teaching scenario tailor-made for the translation industry"' paper presented at the *6th EAMT Workshop, Teaching Machine Translation*, November 14-15, 2002. European Association for Machine Translation, 107-116

Vieira. L. N. and Specia, L. (2011) 'A Review of Machine Translation Tools from a Post-Editing Perspective' 3rd *Joint EM+/CNGL Workshop Bringing MT to the User: Research Meets Translators (JEC 2011),* Luxembourg.

XLIFF and LCX: A Comparison

Asanka Wasala, Dag Schmidtke* and Reinhard Schäler
Localisation Research Centre
Department of Computer Science & Information Systems,
University of Limerick, Limerick, Ireland
***Microsoft European Development Centre (EDC)**
South County Business Park, Leopardstown, Dublin 18, Ireland
{asanka.wasala, reinhard.schaler}@ul.ie, *dags@microsoft.com

Abstract

As the complexity of localisation projects increases, the need for flawless interoperability between different localisation tools becomes ever more important. The XML-based Localisation Interchange File Format (XLIFF) has been developed to address issues related to the exchange, the storing and the transport of localisation data and metadata between different tools across the localisation process. XLIFF is gaining increased acceptance within the localisation community. However, some leading software industry publishers use proprietary file formats rather than the XLIFF standard. This paper discusses XLIFF and compares it to the Localisation Content Exchange (LCX) file format, the internal software localisation file format used by Microsoft. The paper also reviews the main reasons for the lack of acceptance and implementation of the XLIFF standard. It reports on research that investigated the possibility of converting between the LCX and XLIFF file formats and implementing a demonstrator. The paper also describes the schema mapping between the above file formats and the implementation of a prototype converter.

Keywords: *localisation, standards, XLIFF, LCX, interoperability, comparison, schema mapping*

1. Introduction

Standards play an important role in today's localisation industry. Standards enable better interoperability between systems. Some of the prominent standards used in the localisation industry include: XML Localisation Interchange File Format (XLIFF), Translation Memory eXchange (TMX)[1], Term Base eXchange (TBX)[2], Segmentation Rules eXchange (SRX)[3], Portable Object (PO) and Internationalization Tag Set (ITS). Among these standards, XLIFF probably plays the most important role as it has been developed to address interoperability issues between localisation tools. Therefore, in this paper, we mainly focus on experimental research on the XLIFF standard and especially on various interoperability issues associated with XLIFF. Despite the fact that XLIFF was initially developed as an exchange format, it is increasingly being used as an internal file format too. The XLIFF standard is gaining acceptance within the localisation community. However, some leading software publishers use proprietary standards rather than XLIFF, for various historical and other reasons. In this paper, we will compare XLIFF, the open localisation standard by OASIS, with LCX, the internal proprietary localisation standard used in Microsoft.

The paper is organised as follows: The subsections 1.1 and 1.2 present an overview of XLIFF and the LCX file formats; Section 2 briefly discusses the typical usage of the XLIFF and LCX formats. Section 3 discusses findings of the study; Section 4 describes the XLIFF-LCX conversion methodology, while section 5 presents the general discussion. Finally, Section 6 concludes and points to future work.8

1.1 XML Localisation Interchange File Format

XML Localisation Interchange File Format (XLIFF) is an open standard for exchanging localisation data and metadata. It has been developed to address various issues related to storing and exchanging localisation data.

[1,2,3] The TMX ,TBX and SRX formats were developed and maintained by the Localisation Industry Standards Association (LISA). In March 2011, LISA declared bankruptcy and will make its standard openly available under a neutral name.

The XLIFF standard was first developed in 2001, by a technical committee formed by representatives of a group of companies, including: Oracle, Novell, IBM/Lotus, Sun, Alchemy Software, Berlitz, Moravia-IT, and ENLASO Corporation (formerly the RWS Group). In 2002, the XLIFF specification was formally published by the Organisation for the Advancement of Structured Information Standards (OASIS) (XLIFF-TC 2008b, Raya 2004).

The purpose of XLIFF as described by OASIS is to "store localizable data and carry it from one step of the localisation process to the other, while allowing interoperability between tools" (XLIFF-TC 2008b). By using this standard, localisation data can be exchanged between different companies, organisations, individuals or tools. Various file formats such as plain text, MS Word, DocBook, HTML, XML etc. can be transformed into XLIFF, enabling translators to isolate the text to be translated from the layout and formatting of the original file format.

The XLIFF standard aims to (Corrigan and Foster 2003):

- *"Separate translatable text from layout and formatting data;*
- *Enable multiple tools to work on source strings;*
- *Store metadata that is helpful in the translation/localisation process."*

The XLIFF standard is accepted by almost all localisation service providers and is supported by the majority of localisation tools and CAT tools. The XLIFF standard is being continuously developed further by the OASIS XLIFF Technical Committee (TC) (2011).

1.2 Localisation Content Exchange File Format

In order to address the different needs of the Microsoft localisation community[4], Microsoft uses a series of XML files to store software localisation data and metadata. These files, together, are referred to as the Localisation Content Exchange (LCX) container or Localisation Content Exchange file format. LCX evolved as a replacement for a prior, database-driven, software localisation model and toolset, with the main change being a move to store data in XML. The LCX-based architecture provides a programmable Object Model to access localisation content encoded in LCX files programmatically. The benefits of the LCX container (Microsoft 2009) include:

- *"Common localisation project and transportation format;*
- *Diff-able text files instead of a binary format;*
- *Published XML schema;*
- *Data transparency through a complete object model."*

These LCX files are processed by the Microsoft Localization Studio (LocStudio) suite of tools, and other Microsoft software localisation tools. The LocStudio suite consists of several applications and utilities that provide functions ranging from localisation content extraction to localised product building.

2. XLIFF and LCX Usage

2.1 XLIFF Usage

XLIFF eliminates the need for multiple file formats in a localisation workflow. In an XLIFF-based workflow, XLIFF acts as an intermediate file format where all non-XLIFF tool specific file formats will be converted by a pre-processor to XLIFF at a very early stage of the localisation workflow (see figure 1). Then the converted XLIFF files will be handed off to the localisation engineers. It is also noteworthy to mention that there are XLIFF compliant tools capable of producing XLIFF files directly, which will eliminate the need for a pre-processor.

Once tool specific files are converted into XLIFF, XLIFF compliant editors can be used to facilitate the translation process. Having completed the translation process, the XLIFF files will be converted back to native file formats. This workflow simplifies the localisation process by eliminating the need to deal with multiple proprietary file formats throughout the workflow (XLIFF-TC 2008a).

Moreover, XLIFF can also be used as a data container that travels through the workflow (Wasala et al. 2011). The XLIFF file can be populated with various data and metadata during different stages of the workflow. In this scenario, the `<alt-trans>` feature of XLIFF can be utilised to include

[4] Note: for external customers, Microsoft provides XLIFF support in the Multilingual Application Toolkit (MAT), a Microsoft Visual Studio add-in designed to help localize Windows app with translation support, translation file management, and editor tools. This add-in is available available for free download at http://msdn.microsoft.com/en-us/windows/apps/hh848309.aspx.

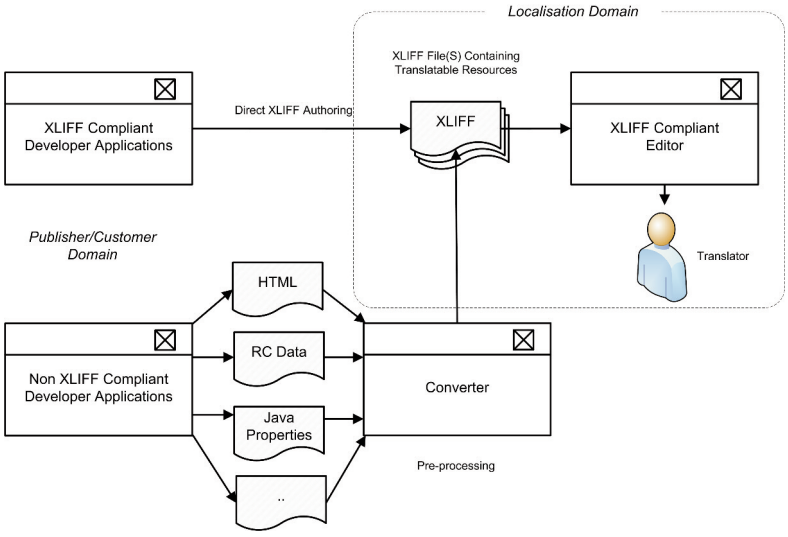

Figure 1: A Simple XLIFF-based Localisation Workflow (Adapted from (XLIFF-TC 2008a))

translations by these CAT tools in an XLIFF file (Wasala et al. 2011, XLIFF-TC 2008a).

Viswananda and Scherer (2004) describe two approaches for using XLIFF in a localisation workflow:

1. Permanent XLIFF files

   In this approach, XLIFF files will be stored in the source control system. The XLIFF files will be converted to native file formats only when building the target files.

2. Transient XLIFF files

   In this approach, localisation content will be stored in the source-control system in native file formats. The native formats will only be converted to XLIFF to exchange with translators.

Out of these two approaches the former approach is recommended by Viswananda and Scherer (2004). However, the selection of the best approach out of the above two will depend on criteria such as richness of the native formats, performance of supporting tools, including the conversion tools, and the data-loss during the round-trip conversion process (Viswananda and Scherer 2004).

2.2 LCX Usage

The LCX based localisation approach consists of seven major steps as depicted in figure 2. A variety of software and file formats are used throughout the process. The major steps are summarised as follows.

1. Generate LCX source files

   This step is carried out at Microsoft and involves the creation of LCX files by processing localisable files (e.g. EXE, DLL, HTML etc). This step also involves the incorporation of developer and localisation pilot team comments, instructions and validation rules to the generated LCX files.

2-4. Preparation and handoff

   The main activity carried out in this phase is the auto-translation. Existing glossary files are used for this purpose. Then an XML based virtual archive containing all of the files required for translation is created. This archive is then handed over to the vendors for translation.

5. Translation

   The translation is carried out by vendors using software called LocStudio. This process also involved dialog-resizing. Finally, validations are performed using the same software based on the information contained in the LCX file, per resource.

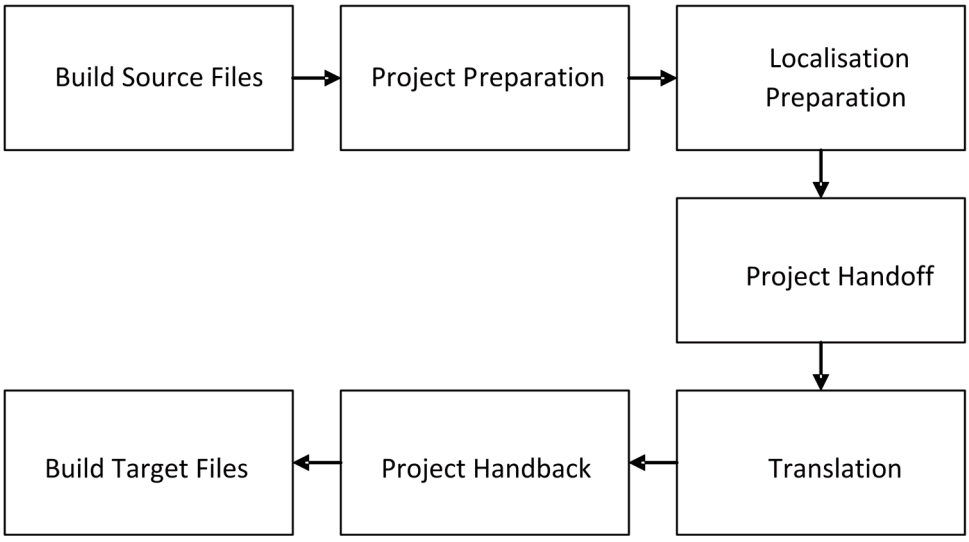

Figure 2: Basic Localisation Workflow at Microsoft

6. Hand-back & validation

   The archive containing the translated resources is then handed back to the Microsoft. The translated resources are extracted and further validations are performed on them prior to the acceptance process.

7. Build target files

   The Final step involves generating localised files (e.g. EXE, DLL, HTML) files from original files and translated LCX files.

3. XLIFF and LCX Format Comparison

One of the main objectives of our research was to compare XLIFF with the LCX format. The first phase involved reviewing official documentation on both the LCX and XLIFF formats. In the second phase, a systematic element-by-element comparison of schemas for LCX and XLIFF was carried out. The third phase involved the manual analysis of sample LCX and XLIFF files that represented localisation data extracted from selected Win32 applications. This section describes the findings of the above analysis. It also presents a comparison of prominent features of XLIFF and the LCX file format.

3.1 Features of XLIFF and LCX formats

Table 1 summarises the results of our comparison. In the following sub sections (sections 3.1.1-4) we describe the advantages, disadvantages, as well as unique features of XLIFF and LCX formats in detail.

3.1.1 Advantages and Unique Features of XLIFF

XLIFF, maintained by the OASIS Technical Committee, is becoming popular in the localisation industry. The localisation industry has realized the advantages and benefits of using XLIFF. XLIFF support is continuously improved in existing tools while new tools are being developed to support XLIFF. Therefore, XLIFF will most likely soon become the most widely accepted localisation data container. XLIFF support in web based systems and services is being increased (e.g. Pootle[5], Drupal[6], Transifex[7] and Lingotek[8]). Therefore, it can be expected that in the near future massive amounts of localisation related data will be available in XLIFF. In order to make use of these resources, localisation tools and services should support this standard. To date, XLIFF is the most complete open standard available to store localisation data and metadata. Furthermore, the XLIFF standard is being improved continuously with the input, support and collaboration of localisation experts.

XLIFF can represent virtually any file format with

[5] http://translate.sourceforge.net/wiki/pootle/index
[6] http://drupal.org/project/xliff
[7] http://help.transifex.com/features/formats.html#xliff-files
[8] http://www.lingotek.com/

	XLIFF	LCX
Advantages & Unique features	• Open and flexible	• Proprietary and controlled
	• Support for variety of resource formats, both binary and textual	• Binary encoded information representation scheme
	• Ability to store supplementary information and metadata	• Precise and secure representation of localisation items and GUI components
	• Inline elements	• Property Bags
	• Alternative translations	• Consistent hierarchy
	• Ability to associate contextual information	• Ability to embed target content validation rules
	• Ability to specify pointers to external resources	• Ability to associate comments with most of the elements
	• Ability to specify segmentation of translations	• Availability of an Object Model
Disadvantages	• Lack of tool support	• Inability to store alternative translations
	• Flexibility of extensions that results in the misuse of extensions	• Limited support for non-binary file formats
	• Extensive use of proprietary data through extensions	• Heavy use of CDATA sections
	• Lack of powerful and accurate mechanism to represent GUI components	• Lack of specific mechanisms to store contextual information
	• Weaknesses of the <note> element	• Lack of mechanisms to represent segmentation information
	Ability to represent the same information using different tags or tag configurations	• Lack of explicit mechanism to store references to external files
	• Lack of support for representing workflow information	
	• Inability to associate metadata related to external resources	

Table 1: Summary of the XLIFF-LCX file format comparison

localisation resources. XLIFF can store both binary resources and textual resources. Therefore, it can be used to represent localisation data from software and program components (e.g. dll, exe) as well as other documentation file formats (e.g. txt, html, xhtml, xml, doc, docx, odf etc). Moreover, XLIFF can be readily used in web services to facilitate the localisation process (Mateos 2010, Wasala et al. 2011). In addition to localisation data, XLIFF can store other supplementary information such as references to glossaries and administrative information such as various metadata related to the localisation workflow. Various metadata and contextual information associated with localisation data can be stored in a self-descriptive hierarchical manner in XLIFF. In some elements of XLIFF, there is a choice for storing localisation data or related supplementary information (e.g. glossaries, segmentation rules) within the element itself (i.e. internally) or to only store a reference (externally) to the actual resource. This provides a handy mechanism to store only pointers to cumbersome files (e.g. bitmap files, TMX files) reducing the complexity of the XLIFF file itself.

In XLIFF, alternative translations corresponding to a particular localisation item can be stored. The ability to store alternative translation brings significant benefits to the translators. Furthermore, there are nine inline elements defined in XLIFF. These elements are useful to accurately retain the formatting (and other) information attached to localisation data especially found in non-binary file formats and content (e.g. to represent bold text found

in an *html* paragraph element). Also, another noteworthy feature of XLIFF is the ability to specify proper segmentation of translation units.

3.1.2 Disadvantages of XLIFF

A common observation noted with regards to XLIFF is that many features described in the XLIFF specification are either not supported or only partially supported by localisation tools (Bly 2010, Imhof 2010, Lieske 2011). The research revealed that there is no localisation tool capable of exporting Win32 resources into XLIFF v1.2. Though most localisation tools are capable of importing XLIFF files, tools capable of exporting localisation content into XLIFF are rare.

XLIFF provides many extension points through the XML namespace mechanism. Some localisation tools use this mechanism to insert vendor-specific proprietary data into XLIFF files (Vackier 2010). This has inevitably led to the creation of different flavours of XLIFF (Lieske 2011). The extensive use of proprietary data in XLIFF files has adverse effects on interoperability (Vackier 2010). For example, although a pre-defined attribute and attribute values for storing the status of a translation unit in XLIFF exists, in different XLIFF flavours (introduced by different tool providers) different tool specific attributes and attribute values are used to denote the translation status of a translation unit. The use of proprietary data and extensions in the above manner diminishes the interoperability of XLIFF content among different tools. In the absence of a mechanism to control or manage custom extensions through the XLIFF specification, tools will continue to freely use such extensions.

According to the XLIFF specification (XLIFF-TC 2008b), "*The <x/> element is used to replace any code of the original document"*. Since 'any code' can be replaced by using this element, it can be used to store tool specific information (i.e. proprietary data). Consider the following XLIFF mark-up segment:

```
<trans-unit id="1729" translate="no">
      <source>
            <x id="79"/>
            <x id="80"/>
      </source>
</trans-unit>
```

In the sample segment, although the <trans-unit> element has been used, no translatable content is available. However, a generic placeholder is used as a stand-alone element within the <source> element to store tool-specific data or to reference tool specific data. Although an XLIFF file with content similar to the above mark-up can be perfectly valid, the information is not decodable by tools other than the one used to generate it. This use of inline elements significantly affects the interoperability of XLIFF data.

XLIFF does not provide[9] a powerful mechanism to represent GUI components attached to localisation items. However, such information is very useful for visual editors. A typical scenario is to visualise the localisation resources of a Win32 executable file in a visual XLIFF editor. According to the current XLIFF specification (version 1.2), information needed to visualise GUI components is stored in a textual format using attributes defined mainly in <group> and <trans-unit> elements. However, this methodology does not provide an accurate, secure and complete mechanism to represent modern GUI components such as Microsoft .NET dialogs, Silverlight components, Adobe Flash Components, wxWidgets etc.

Another disadvantage of XLIFF is the inability to group or further categorise notes or comments. Moreover, <note> elements cannot be extended by including non-XLIFF attributes or elements. In addition, the usage of the <note> element is restricted to some elements (e.g. a <note> element common to both <header> and <body> elements cannot be associated with a <file> element; <note> elements cannot be associated with individual <source> elements or <target> elements). Due to these weaknesses of the <note> element, it is difficult to map similar elements in other localisation file formats to XLIFF's <note> element in a lossless manner.

The same information can be represented in different ways using XLIFF (Lieske 2011, Bly 2010). This makes the implementation of XLIFF in tools complex and difficult. Especially, the grouping of localisable items, segmentation methodology and inline element usage can be different in different tools.

An XLIFF file might travel through different phases and be used by different tools in a localisation workflow. Although XLIFF has a mechanism to describe phases and tools involved, there is no way to

[9] It could be argued that it would be difficult, if not impossible, for any open exchange file format such as XLIFF to cater for the wide variety of GUI representations available, which makes it not a limitation of XLIFF, but of the approach taken by the XLIFF community.

track the status of a certain phase in XLIFF, nor is there a mechanism to track the status of a tool (i.e. whether a tool has already performed an action, or is still waiting to perform an action). The only solution to address this problem would be to extend the XLIFF schema to include this information. However, that will certainly affect the interoperability (Wasala et al. 2011).

In XLIFF, references to external files or internal files can be stored. However, these references cannot be further described using metadata. Moreover, the current XLIFF specification does not allow embedding XML content as an internal file reference (Wasala et al. 2011).

3.1.3 Advantages and Unique Features of LCX

The LCX format relies heavily upon binary information[10], mainly to preserve the formatting related information of localisation items and to accurately represent GUI components. This binary encoded representation scheme provides an accurate, secure and powerful representation mechanism for resources.

The LCX format provides an object model, which defines how to programmatically access and manipulate LCX content. Therefore, the development of LCX based tools is easy. LocStudio and other tools that make use of the LCX object model can directly consume and load appropriate external visual editors for binary resource streams embedded in LCX. LCX content is always generated by the LocStudio suite of software, or by other tools making use of the LCX programmable object model. LCX files cannot be generated manually. Therefore, LCX content, the data representation structure and hierarchy are always consistent in the LCX format.

The LCX format has a unique controlled extensibility mechanism through its Property Bags feature. Property Bags can be associated with most LCX format elements. A Property Bag can store several *key*, *value* pairs. The *keys* can be defined to hold *values* in different data types including Numbers (Single and Int32 types), Boolean, Strings, and Dates. Property Bags are useful in storing various properties of localisation items as well as metadata. Moreover, the LCX format provides a mechanism to embed target content validation rules along with the localisation content. Therefore, the target content validation can be carried out by translators themselves.

In LCX it is possible to associate comments with most elements. The comments can be grouped and they can be either enabled or disabled.

3.1.4 Disadvantages of LCX

One of the major drawbacks of LCX is its inability to store alternative translations for localisable elements. Alternative translations are very useful for translators, especially in the absence of resources such as translation memories (TM), glossaries etc. Moreover, translators can also get an idea about the proper translation for a localisable item by looking at different translations of the same item in other languages.

There are limitations in the support for non-binary file formats such as HTML, XML etc. The LCX format is designed for storing localisable content from software resource file formats (e.g. Win32 resources, DLL etc). Although it can represent the localisation content of some non-binary file formats, the support for such files is minimal when compared to XLIFF. Specifically, LCX does not have any inline elements. Therefore, it is not as well suited for the handling of complex tagged content, such as localisable items of an HTML web-page. In Microsoft, the content localisation of HTML, XML and other content file formats is handled by dedicated content localisation tools, not by LocStudio.

Access to contextual information related to a localisation item would be very useful for translators (Sikes 2011). However, in LCX no specific elements are there to store contextual information. Contextual information related to localisation items is usually stored as comments.

LCX does not provide a mechanism to represent a proper segmentation methodology for text. Furthermore, the line breaks included within localisable text are encoded in binary. This is a reflection of the legacy and primary use of LCX for software localisation, where text segmentation is not a primary concern. The LCX representation of an example sentence that includes two line breaks is given in example 1. In example 1, `A;` represents a line break in LCX format.

In LCX, there is no explicit mechanism to store references to external files. Moreover, although the use of the CDATA sections is not recommended in localisation file formats (XLIFF-TC 2006), LCX

[10] In constrast to the open XLIFF approach, the Microsoft proprietary LCX is fine tuned to represent Microsoft-specific GUI components.

```
<Str Cat="Text">
  <Val><![CDATA[Cannot open the %% file.]A;]A;Make
  sure a disk is in the drive you
  specified.]]></Val>
</Str>
```

Example 1

relies heavily upon CDATA sections to store localisation data. Therefore, special care has to be taken when programmatically manipulating LCX content.

4 XLIFF ↔ LCX Conversion

Our core research interest is to propose a mapping between XLIFF - LCX and to identify mapping issues.

The XLIFF and LCX format comparison revealed incompatibilities between the two file formats. We identified some LCX specific data that cannot be successfully mapped to XLIFF without extending the XLIFF Schema to represent LCX data. Therefore, we propose to extend XLIFF to represent both XLIFF and LCX syntax. We call the new format LCX-XLIFF for the purpose of our research. The LCX-XLIFF format complies with the XLIFF 1.2 transitional schema. We propose to use the LCX-XLIFF format as an intermediate data container between LCX and XLIFF in the conversion process as it minimises the data loss during the roundtrip conversion process (see figure 3). The research only focuses on the conversion between LCX and LCX-XLIFF formats.

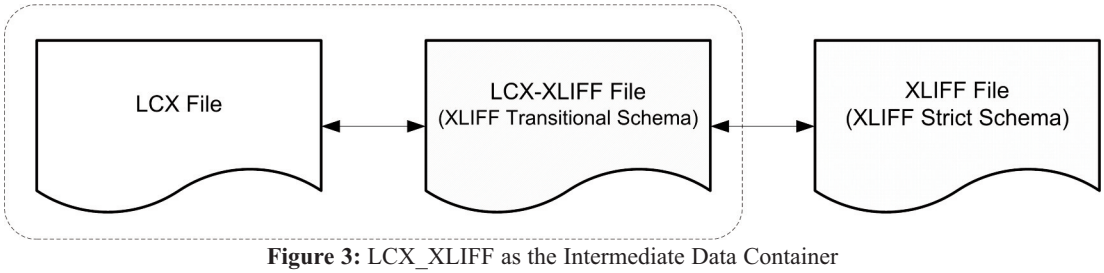

Figure 3: LCX_XLIFF as the Intermediate Data Container

4.1 Proposed Methodology

n this research, we propose two different ways of representing LCX content in XLIFF. The two different approaches are categorised as "*maximalist*" and "*minimalist*" representation approaches.

The main idea behind the maximalist approach is to decode as much as possible of the information contained within LCX and represent it in XLIFF with few or no extensions. In this approach, information needed to visualise User Interface (UI) resources in localisation tools will be decoded from LCX and represented in XLIFF. For example, binary information such as coordinates of UI controls in LCX will be decoded and represented in XLIFF in textual format using appropriate tags. Information on accelerators (keyboard shortcuts) will also be decoded and represented in a textual format in XLIFF. Other binary resources included in LCX such as bitmaps, icons etc. will be represented in XLIFF's binary translation units. Prior to representing textual information in XLIFF, proper segmentation is carried out on the source LCX textual elements. Then, the information with the proper segmentation guidelines is stored in the XLIFF documents. Furthermore, textual information included in the LCX's CDATA sections will be carefully analysed and represented in XLIFF as entity references, inline elements and Unicode characters. LocStudio specific metadata (e.g. `<Disp>`, `<Modified>` elements) will be retained by extending XLIFF using the namespace mechanism. This approach minimises the data loss while maximising the interoperability of converted XLIFF. The maximalist approach is highly recommended for the LCX to XLIFF conversion process. However, the complexity of the conversion process is high in this approach.

In the minimalist approach, binary encoded data in LCX will not be decoded at all. Hence, only the textual localisation data will be included in XLIFF. However, all binary and other LCX specific information will be retained within XLIFF. This is achieved by extending the XLIFF namespace mechanism. Converted XLIFF files using this

approach can then be used with different tools as well as with translation memories. Although it will be possible to translate textual content in the converted file, it will not be possible to visualise UI resources and make changes to the UI by using LCX-XLIFF files converted using the minimalist approach.

As an example, consider the LCX content in example 2 which corresponds to the resource illustrated in figure 4:

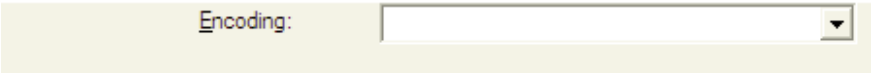

Figure 4: Sample UI Resource

```
<Item ItemId="259" ItemType="130;WIN_DLG_CTRL_" PsrId="3" Leaf="true">
        <Str Cat="Static Text">
          <Val><![CDATA[&Encoding:]]></Val>
        </Str>
        <Bin BinId="9">

<Val><![CDATA[AQAAAAAAAAAAAAAAAFBEAAEAKAAoAAAAAAAAAAABggAAAA==]]></Val>
        </Bin>
        <Disp Icon="Dlg"/>
      </Item>
```

Example 2

The expected output of the conversion of the above LCX content into XLIFF using the maximalist approach is shown in example 4:

The scope of this research was limited to the minimalist approach. An element by element mapping is proposed to convert between XLIFF and LCX in the minimalist approach.

4.2 Prototype Implementation

A prototype was implemented to demonstrate the conversion between the LCX and LCX-XLIFF file formats.

```
<trans-unit id = '206' resname = '259' restype = 'static' style =
'0x50000000' coord = '68;1;40;40' lcx:ItemId="259"
lcx:ItemType="130;WIN_DLG_CTRL_" lcx:PsrId="3" lcx:Leaf="true">
  <source>&amp;Encoding:</source>
  <lcx:Disp Icon="Dlg"/>
</trans-unit>
```

Example 3

The expected output of the conversion of the same LCX content into XLIFF using the minimalist approach is shown in example 3:

XSL Transformations (XSLT) were developed for mapping between the LCX and LCX-XLIFF formats. Using an XSLT processor (such as Saxon) or an

```
<trans-unit id="d0e462" resname="259" lcx:ItemId="259"
lcx:ItemType="130;WIN_DLG_CTRL_" lcx:PsrId="3" lcx:Leaf="true">
   <source>&amp;Encoding:</source>
     <lcx:Str Cat="Static Text">
     <lcx:Val><![CDATA[&amp;Encoding:]]></lcx:Val>
     </lcx:Str>
     <lcx:Bin BinId="9">
<lcx:Val><![CDATA[AQAAAAAAAAAAAAAAAFBEAAEAKAAoAAAAAAAAAABggAAAA==]]>
</lcx:Val>
         </lcx:Bin>
         <lcx:Disp Icon="Dlg"/>
</trans-unit>
```

Example 4

XML processing tool, the conversion can be carried out. However, to facilitate the conversion, a separate tool was developed. The tool is capable of converting LCX files (known as LCL[11] files) generated by the LocProject Editor into LCX-XLIFF and LCX-XLIFF files into LCX format. A general software localisation workflow that involves the aforementioned converter is depicted in figure 5

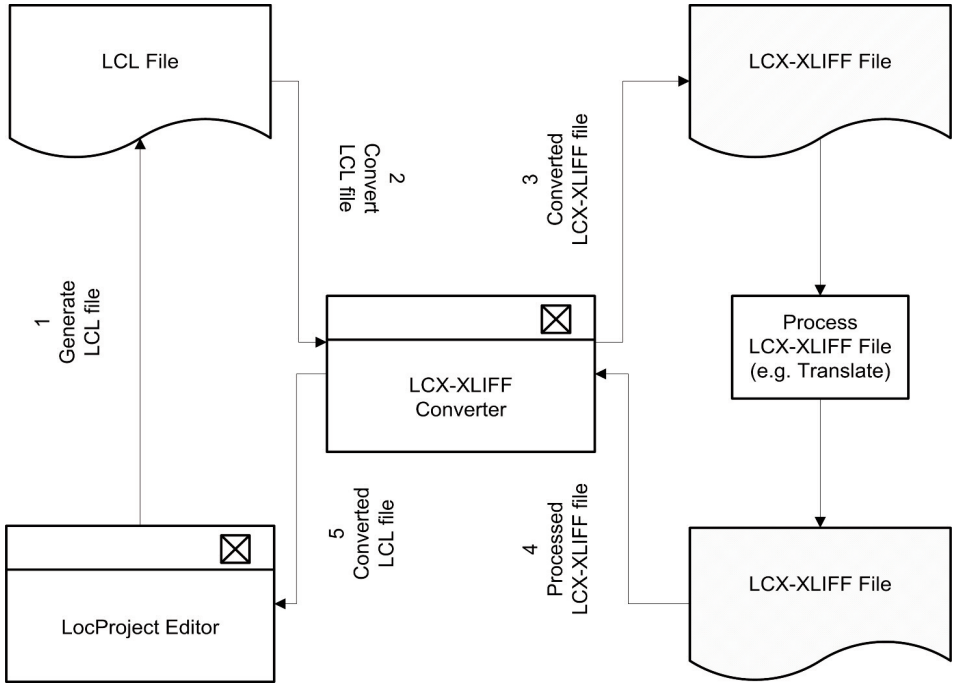

Figure 5: Typical Localisation Workflow with LCX-XLIFF Converter

The workflow in figure 5 involves five distinct steps:

Step 1: The LocProject Editor is used to generate a LCL file corresponding to a resource to be localised.

Step 2: The LCL file is converted to LCX-XLIFF by using the converter.

Step 3: The converted LCX-XLIFF file can be processed at this stage using third party XLIFF enabled tools. The translation process can be carried out using the LCX-XLIFF file.

Step 4: The processed LCX-XLIFF file is then converted back to LCL using the converter.

Step 5: The processed LCL file can then be opened for further processing, inspecting, validation etc. using the LocProject Editor or LocStudio application.

5 Discussion

5.1 XLIFF & Interoperability Issues

The XML-based Localisation Interchange File Format (XLIFF) has been developed, mainly to, address issues related to the exchange of localisation data and metadata between different tools across the localisation process. Perhaps surprisingly, the adoption of the XLIFF standard is still somewhat limited. Our research indicates that there are three main reasons for lack of adoption of the XLIFF standard:

1. Lack of awareness about XLIFF (Anastasiou 2010);
2. Limitations of the standard;

For example, vaguely defined criteria (Bly 2010) in the XLIFF specification confuse tool developers. In addition, tools providers face various difficulties in the absence of proper metadata storage and usage conventions. Due to the extremely flexible nature of XLIFF, tool providers have come up with different flavours of XLIFF (Lieske 2011, Imhof 2010,

[11] LCL is a version of the LCX schema specifically for localisable text.

Anastasiou 2010).

3. Improper or incomplete implementation of the standard.

Our research revealed that many XLIFF features are either not supported or only partially supported by tools (Anastasiou 2010, Bly 2010, Lieske 2011). This seems to be due to mainly two reasons:

1. Business Case Requirements

Depending on the requirements of different business cases, different tools have been implemented to support different parts of the XLIFF specification. There are localisation tools that do not support XLIFF at all, which is mainly due to the fact that there is no strong business case demanding XLIFF support from these tool vendors.

2. Complexity and limitations of the standard

Although XLIFF's formal tool compliance is easy to achieve, complete XLIFF feature implementation in tools is difficult due to the complexity of the standard (Anastasiou and Morado-Vázquez 2010). Moreover, the segmentation and inline element usage is only vaguely defined in the specification. Therefore, tools can represent the same information in different ways in XLIFF. This greatly affects the interoperability.

Therefore, although many tools claim that they are XLIFF compliant, many XLIFF features are missing in those tools. The actual level of XLIFF compliance in localisation tools is still an open question. The XLIFF-TC (2011) is working on introducing an XLIFF conformance clause to address these issues. Some of the observations related to XLIFF support in tools are summarised as follows:

- Cannot export localisation content into XLIFF v1.2 format
  - -Especially Win 32 content
- Inability to open XLIFF with multiple file elements
  - -e.g. Alchemy Catalyst 8, SDL Passolo 2009
- Usage of different source/target language formats
  - -e.g. en-US, US
  - Imhof (2010) and Bly (2010) give further examples:
- Support for Alternative Translation Units
  - -e.g. MemoQ, Trados 2009 do not support the `<alt-trans>` element
- The use of custom defined values to denote the status of the translation
  - -e.g. Trados 2009, MemoQ
- Support for translate attribute and translate state attributes
- Support for `<note>` element
  - -e.g. MemoQ, Trados 2009 do not support the `<note>` element

The lack of tool support for the XLIFF standard, the complex nature of the standard and internal flaws of the standard greatly affect its ability to deliver on the interoperability promise.

5.2 XLIFF ↔ LCX Comparison

The comparison of XLIFF and LCX revealed that both file formats have unique advantages and disadvantages. The unique features of each file format are found to be closely related and are often complementary to each other. However, it is difficult to achieve 100% compatibility due to factors such as the extensive use of binary data in LCX format, inclusion of tool specific data in LCX and inline element usage in XLIFF. A file format converter however, similar to the one developed in this research, can provide an acceptable level of interoperability between the two file formats.

Furthermore, it was evident that both the XLIFF and LCX formats offer much more functionality than just localisation data exchange. These file formats have morphed into, and are being used as, data repositories and storage formats for localisation knowledge. They are now being specifically designed for data maintenance and reuse. Moreover, both file formats have been designed to accommodate efficient workflows.

Based on the results and analysis, improvements to the XLIFF and LCX formats are proposed.

5.2.1 Proposed improvements to XLIFF

- A binary representation of user interface (UI) elements (as in LCX format)

- The ability to associate the `<note>` element with any element and to further customise or categorise the `<note>` element (this issue has been identified already and will be implemented in XLIFF version 2.0 (XLIFF-TC 2011).

- A mechanism similar to LCX's Property Bags in XLIFF to store metadata related to each localisable item. This feature would be useful in defining a localisation memory container using

XLIFF.

- A mechanism to embed target validation rules.
- Microsoft uses a special file container named LSPkg (Localisation Studio Package) to store several related LCX files (as well as related other files) within a single file. An LSPkg equivalent data container in XLIFF would be useful to store several file elements in a single file. Information common to several file elements (e.g. various settings etc.) could be stored within this container, e.g. a zipped data container (similar to Microsoft DOCX format). This would allow more data to be stored in an efficient manner. This compact data container would be easier to exchange between different systems. It would further serve the purpose of a localisation memory container.
- The ability to further describe internal and external references would be useful (e.g. to include an attribute to describe the type of reference).

5.2.2 Proposed Improvements to the LCX file format

- Inline elements.
- The ability to store references to external files/data sources (e.g. glossaries, translation memories, bitmaps etc.). There should be a mechanism to describe the references with the use of metadata.
- The ability to store alternative translations.
- The ability to store contextual information of a certain localisation item.
- The ability to specify segmentation.
- XML canonicalisation (W3C, 2001) would improve the interoperability of the LCX file format.

6. Conclusions and Future Work

In this paper we compared the XLIFF and the LCX file formats and identified interoperability issues between them. Our work revealed prominent issues associated with the XLIFF and the LCX standards.

A prototype converter based on XSLT was developed to demonstrate the conversion between XLIFF and LCX formats. However, due to the complex hierarchy of `<Item>` elements in LCX and the limitations of XSLT processing (e.g. issues associated with CDATA processing, white space preservation etc.) XSLT style sheets were found to be inefficient and inappropriate for the development of an LCX to XLIFF converter. A programmatic approach that utilizes the LCX Object Model is recommended for the implementation of a practical converter.

A formal evaluation of the LCX ↔ XLIFF round-trip conversion process should be undertaken as part of a follow-up study by using a set of experimental data as well as real data of the type used in a typical localisation workflow.

This research confirmed the need for further research into standards compliance, conformance and interoperability of localisation tools and technologies. Especially to assess the level of implementation of standards in various localisation tools and to evaluate actual usage of standards in different localisation workflows.

Acknowledgements

This work has been supported by a grant from Microsoft European Development Centre, Ireland.

References

Anastasiou, D. (2010) 'Survey on the Use of XLIFF in Localisation Industry and Academia', in *7th International Conference on Language Resources and Evaluation (LREC)* Malta.

Anastasiou, D. and Morado-Vázquez, L. (2010) 'Localisation Standards and Metadata' in Sánchez-Alonso, S. and Athanasiadis, I. N., eds., *Metadata and Semantic Research*, Springer Berlin Heidelberg, 255-274.

Bly, M. (2010) 'XLIFF: Theory and Reality: Lessons Learned by Medtronic in 4 Years of Everyday XLIFF Use', in *1st XLIFF Symposium*, September 22, 2010, Limerick, Ireland.

Corrigan, J. and Foster, T. (2003) 'XLIFF: An Aid to Localization', available: http://developers.sun.com/dev/gadc/technicalpublications/articles/xliff.html. [accessed 10-06-2009].

Imhof, T. (2010) 'XLIFF – a bilingual interchange format', in *MemoQ Fest*, Budapest, Hungary, 5-7 May 2010.

Lieske, C. (2011) 'Insights into the future of XLIFF', *MultiLingual,* 22(5), 51-52.

Mateos, J. (2010) *A Web Services approach to software localisation. Bringing software localisation tools from the desktop to the cloud*, thesis (Masters), University of Dublin.

Microsoft (2009) *LSWiki for LocStudio 6.x*, unpublished.

Raya, R. (2004) 'XML in localisation: A practical analysis', available: http://www.ibm.com/developerworks/xml/library/x-localis/. [accessed 10-06-2009].

Sikes, R. (2011) 'The New MultiCorpora TMS', *ClientSide News*, 13-15, available: http://www.clientsidenews.com/downloads/CSNV11I1.pdf [accessed 21-06-2012].

Vackier, T. (2010) 'The Double-Edged Sword of Extensibility in XLIFF', in *1st XLIFF Symposium*, Limerick, Ireland.

Viswananda, R. and Scherer, M. (2004) *Localizing with XLIFF and ICU,* translated by San Jose, California USA: IBM Corporation.

W3C (2001) 'Canonical XML Version 1.0', available: http://www.w3.org/TR/xml-c14n [accessed 21-06-2012].

Wasala, A., O'Keeffe, I. and Schäler, R. (2011) 'Towards an Open Source Localisation Orchestration Framework ', *Tradumàtica,* 9, 84-100.

XLIFF-TC (2006) 'XLIFF 1.2 Representation Guide for HTML', available: http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html [accessed 08-06-2009].

XLIFF-TC (2008a) 'OASIS XML Localisation Interchange File Format TC - Frequently Asked Questions', available: http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html [accessed 08-06-2009].

XLIFF-TC (2008b) 'XLIFF 1.2 Specification', available: http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html [accessed 08-06-2009].

XLIFF-TC (2011) 'XLIFF 2.0', available: http://wiki.oasis-open.org/xliff/XLIFF2.0 [accessed 02-09-2011].

Guidelines for Authors

Localisation Focus
The International Journal of Localisation
Deadline for submissions for VOL 12 Issue 1 is 30 June 2013

Localisation Focus -The International Journal of Localisation provides a forum for localisation professionals and researchers to discuss and present their localisation-related work, covering all aspects of this multi-disciplinary field, including software engineering and HCI, tools and technology development, cultural aspects, translation studies, human language technologies (including machine and machine assisted translation), project management, workflow and process automation, education and training, and details of new developments in the localisation industry.

Proposed contributions are peer-reviewed thereby ensuring a high standard of published material.

If you wish to submit an article to Localisation Focus-The international Journal of Localisation, please adhere to these guidelines:

- Citations and references should conform to the University of Limerick guide to the Harvard Referencing Style
- Articles should have a meaningful title
- Articles should have an abstract. The abstract should be a minimum of 120 words and be autonomous and self-explanatory, not requiring reference to the paper itself
- Articles should include keywords listed after the abstract
- Articles should be written in U.K. English. If English is not your native language, it is advisable to have your text checked by a native English speaker before submitting it
- Articles should be submitted in .doc or .rtf format, .pdf format is not acceptable
- Excel copies of all tables should be submitted
- Article text requires minimal formatting as all content will be formatted later using DTP software
- Headings should be clearly indicated and numbered as follows: 1. Heading 1 text, 2. Heading 2 text etc.
- Subheadings should be numbered using the decimal system (no more than three levels) as follows:

  Heading
  1.1 Subheading (first level)
  1.1.1 Subheading (second level)
  1.1.1.1 Subheading (third level)
- Images/graphics should be submitted in separate files (at least 300dpi) and not embedded in the text document
- All images/graphics (including tables) should be annotated with a fully descriptive caption
- Captions should be numbered in the sequence they are intended to appear in the article e.g. Figure 1, Figure 2, etc. or Table 1, Table 2, etc.
- Endnotes should be used rather than footnotes.

More detailed guidelines are available on request by emailing LRC@ul.ie or visiting www.localisation.ie

Localisation Focus
The International Journal of Localisation
VOL. 11 Issue 1 (2012)

CONTENTS