Productivity vs Quality?

A pilot study on the impact of translation memory systems

Lynne Bowker, School of Translation and Interpretation, University of Ottawa, Canada

Translators working in the localisation industry are faced with the task of producing high-quality translations in a very short turnaround time. One way in which they are trying to balance these goals is by using translation technology to help. One of the most popular translation technology tools available is the translation memory (TM). TMs are often promoted as tools that can help translators to improve their productivity, but less is known about the impact that the use of such tools can have on the quality of the translation. This article investigates the impact of TMs on both speed and quality by presenting the results of a pilot study in which three groups of student translators were asked to translate the same text. Translators in Group A did not use a TM, and while their translations were of relatively high quality, they took longer to produce. Translators in Group B used an “unadulterated” TM, and they were able to translate more quickly, but there were some minor concerns with the quality of their work. Translators in Group C used a TM that had been deliberately “seeded” with a number of translation errors, and while they were able to work quickly, the quality of their translations was lower than that of the other two groups. The results of this small experiment seem to indicate that, when faced with the pressure to translate quickly, translators using TMs may not be critical enough of the proposals offered by the system. This in turn indicates a definite need for proper training in the appropriate use of translation technology.

1. Introduction

One of the most talked-about translation tools to enter the market in recent years is the translation memory (TM) (Austermühl 2001; Bowker 2002; Esselink, 2000; L’Homme 1999; Somers 2003). A TM is essentially an aligned bilingual database containing texts that have been previously translated. It is based on the principle of “recycling” previously translated documents – a translator should be able to re-use parts of texts that have been previously translated, and should never have to re-translate a portion of text that has already been translated (O’Brien 1998). An oft-cited reason for using a TM is that it can help to increase a translator’s productivity – with the help of such a tool, a translator can translate a greater volume of material in a given period of time.

This desire to increase productivity is needs-driven. In this era, more and more companies are trying to sell their products in foreign markets – a strategy known as globalisation. This has been greatly facilitated by the Web and e-commerce. Nevertheless, while the Web makes it possible to reach international audiences quickly and easily, research has shown that in order for sales and marketing to be effective, the advertisements, the products themselves, and any accompanying documentation should be presented in the language of the target market.1 For example, according to IDC’s eWorld 2001 survey2, most non-English-speaking Internet users prefer Web sites in their own language. Meanwhile, a survey by Forrester Research3 indicates that business Web users are three times more likely to buy when addressed in their own language. Therefore, in order to stay competitive and maximize profits, many companies are localising their products. The term localisation refers to the process of customizing or adapting a product (e.g. a software package, a Web site) for a target language and culture.

Localisation has grown into a big business (Brooks 2000; Thibodeau 2000). Market watchers at Allied Business Intelligence suggest that the value of revenues from software localisation could reach US$3.4 billion by 2007, while those from Web page localisation could reach US$3.1 billion by the same date (ABI 2002).

This trend towards localisation has resulted in a significant increase in the volume of material to be translated, which has been accompanied by an increase in pressure on translators to work more quickly while still maintaining high quality output. The deadlines that translators are expected to meet have also grown shorter. This is in part because companies want to get their products onto the shelves in all corners of the world as quickly as possible, and the practice of “simship” or simultaneous shipment means that they want to release all language versions of that product at the same time, or within a reasonably short period of time. In addition, electronic documents such as Web sites may have content that needs to be updated frequently (Cheng 2000). Companies want to be sure that their sites reflect the latest information, so translators are under pressure to work very quickly to ensure that the up-to-date information is reflected in all language versions of the site.

Translators are therefore faced with a dilemma – how can they translate faster but still maintain a high standard of quality? Maintaining high quality is essential since a poor translation could result in, at best, wasted time and money, and at worst injury, illness or loss of life (e.g. if medical or maintenance information is mistranslated). One way in
which translators working in the localisation industry are trying to achieve this balance is by turning to technology, such as translation memories (TMs), for assistance.

It is this question of balancing speed and quality with the help of a TM that I wish to examine in this article. As previously mentioned, there have been numerous investigations reporting that TMs can help to improve productivity. By reusing parts of previous translations, translators are bound to save time. However, there have been fewer investigations into the effects that the use of this type of technology has on the quality of the resulting translation.

This paper will be divided into six main sections. Following this introduction, section 2 will briefly explain how a TM works. Section 3 will explore the impact on productivity and quality as observed by other scholars and practitioners. Section 4 will outline a small pilot study undertaken to explore the balance of speed and quality obtained by translators using TMs. Section 5 will discuss the results of this experiment. Finally, section 6 will offer some concluding remarks.

2. How a Translation Memory Works

Translation memory tools are computer-aided tools, which means they are designed to help (rather than replace) human translators. Although they were conceived of as early as the 1970s (Melby 1995), such tools have only been widely commercially available since the late 1990s. As stated above, a TM is a database that contains texts that have already been translated by a human translator.

The data contained in a TM is organized in a very precise way. There are two types of texts stored in a translation memory: source texts, which are the original texts in language A, and target texts, which are the texts that have been translated into language B. In an initial step, a conventional TM tool divides each text into small units known as segments. These segments usually correspond to sentences or sentence-like units (e.g. list items, table cells, titles, headings). Through a process known as alignment, the segments from the source texts are linked to their corresponding segments in the target texts. An aligned pair of segments is known as a translation unit (TU), as illustrated in Table 1.

Table 1. French and English segments are aligned to created TUs

| Translation Unit 1 | FR: L’École de traduction sollicite des candidatures pour un poste menant à la permanence au rang de professeure(e) adjoint(e). | EN: The School of Translation invites applications for a tenure-track position at the rank of Assistant Professor. |
| Translation Unit 2 | FR: La création de ce poste est liée à l’autorisation budgétaire de l’Université. | EN: This position is subject to budgetary approval by the University. |

This alignment can be carried out in one of two ways. Previously completed translations can be automatically aligned after the fact in a process known as post-translation alignment. This is done using alignment software that comes with the TM package. Alternatively, translators can choose to align the translation they are currently working on as they go along.

When translators receive a new text to translate they begin by opening this text in the TM environment. The TM tool proceeds to divide this new text into segments. Once this has been accomplished, the tool starts at the beginning of the document and compares each segment to the contents of the TM database. If it finds a segment that it “remembers” (i.e., a segment that matches one that has been previously translated and stored in the TM database), it retrieves the corresponding TU from the database and shows it to the translator, as illustrated in Table 2. Now translators can refer to the previous translation and adopt or modify it for use in the new translation.

Table 2. An exact match located in a TM

<table>
<thead>
<tr>
<th>Segment from new source text</th>
<th>Matching TU retrieved from TM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Les candidats et candidates doivent être bilingues, anglais-français, et détenir un doctorat dans le domaine de la traduction ou dans une discipline apparentée.</td>
<td>FR: Les candidats et candidates doivent être bilingues, anglais-français, et détenir un doctorat dans le domaine de la traduction ou dans une discipline apparentée.</td>
</tr>
<tr>
<td>candidates should be bilingual, French-English, and hold a PhD in translation or in a related discipline.</td>
<td>EN: Candidates should be bilingual, French-English, and hold a PhD in translation or in a related discipline.</td>
</tr>
</tbody>
</table>

Of course, language is dynamic, which means that the same idea can be expressed in a number of different ways (e.g. ‘Applications should be sent to’ / ‘Please forward your application to’). Consequently, translators cannot always expect to find a high number of exact matches for previously translated segments in the TM. However, it is highly likely that there will be segments in a new source text that are similar to, but not exactly the same as, segments that are already stored in the TM. For this reason, TMs also employ a feature known as fuzzy matching. As shown in Table 3, a fuzzy match is able to locate segments in the memory that are an approximate or partial match for the segment in the new source text. These types of matches can be very useful for translators because at least part of the previous translation may be reusable.
Most TM systems also operate in conjunction with an associated term base, which is essentially an electronic glossary that has been created by a translator using a terminology management system (TMS) that is compatible with the TM. As described above, a TM system compares the new source text segments against the previously translated segments stored in the TM database. At the same time, using a process known as active terminology lookup, the TMS compares the individual terms contained in each source text segment against the terms contained in the termbase. If a term is recognized as being in the termbase, the translator’s attention is drawn to the fact that an entry exists for this term, and the translator can then view the term record and copy and paste the term from the record directly into the target text. This means that, even in cases where no exact or fuzzy matches are found for source text segments, the translator might at least find some translation equivalents for individual terms in the TMS’s termbase.

Nevertheless, there is still a level of linguistic repetition that falls between full sentences and specialized terms – repetition at the level of expression or phrase. For many types of texts, this is the level where linguistic repetition will occur most often. Until recently, most TMs permitted phrase or expression searching only through a “manual” concordancer; in other words, a translator could manually select an expression and instruct the TM to search through the database to find examples. In more recent versions of a number of TMs, however, an auto-concordance function has been added, which, when activated, will automatically search for text fragments when no segment-level match is found.

Once the translator is satisfied with the translation for a given segment – which can be taken directly from the TM database, adapted from a proposed match, or created by the translator from scratch – the newly created TU can be added to the TM database and the translator can move on to the next segment. In this way, the database grows as the translator works.

### 3. Impact of TMs on Productivity and Quality

Because TMs have only been widely available since the late 1990s, it is only recently that people are beginning to learn about the impact these tools can have on the speed and quality of translation. This section will survey some of the observations and speculations that have been made to date. Before doing so, however, it must be emphasized that TM system performance is dependent on the scope and quality of the existing TM database and is expected to improve as the database grows. A TM is of no use if it is empty (i.e., if no translations have been stored in it), and the quality of the translations stored in the TM is dependent on the skill of the translator. Remember that a TM is a tool that is designed to support translators, but the translations suggested by the TM are in fact texts that have been previously translated and stored in the database by human translators.

Another important point to keep in mind is that not all texts are equally suitable for inclusion in a TM. Given that the aim of a TM is to allow translators to reuse previously translated work, it makes sense that the types of texts that are best suited for working with a TM are those which are repetitive or which will be updated or revised. Texts that are written about highly specialized subjects are also good candidates for inclusion in a TM, particularly if the subject matter and format of the texts do not vary greatly and if the translator using the TM frequently works in that subject field.

The most widely professed advantage of a TM is that it will save translators time. Translators who are able to reuse portions of a previous translation will be able to translate texts more quickly, and by increasing their productivity, they will, in principle, be able to earn more money. Webb (1998) conducted a number of surveys and case studies which demonstrated that, when used in the right circumstances (e.g., with suitable text types, with sufficient training), TMs can indeed lead to a significant increase in productivity.7 O’Brien (1998:119) states “our experience has shown that anything from 10% to even as high as 70% can be leveraged from translation memories.”8 Meanwhile, according to Somers (2003:42), a 60% productivity increase may be possible on occasion, but a 30% productivity increase is a more reasonable average expectation.

Following claims of increased productivity, the second most often cited benefit of working with a TM system is that it improves translation quality by increasing consistency. A translator who is working on a long document is able to maintain consistency throughout the text. In addition, many TM systems can be networked, which means that several translators working on the same document or working for the same agency or client can share the same TMs, thereby maintaining consistency within the group. While consistency is often a good thing, it is worth remembering that there are some situations where consistency may not be desirable.9

In addition to the question of consistency, other quality-related issues have been raised by translators working with TMs. One of the most significant is the fact that TM databases store isolated segment pairs, rather than complete texts. In the words of Arrouart and Bédard (2001:30), a TM is actually a memory of sentences out of context.

This can be problematic because the sentences in a text generally depend on each other in various ways. For example, when we read/write the third sentence in a text, we can refer back to information already presented in the first two sentences, which means that it is possible to use pronouns, deictic and cataphoric references, etc. However, if we take that third sentence in isolation, the antecedents of such references may not be clear.

<table>
<thead>
<tr>
<th>Table 3. A fuzzy match retrieved from the TM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment from new source text</td>
</tr>
<tr>
<td>Fuzzy match retrieved from translation memory</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
In addition, because languages do not have a one-for-one correspondence or the same stylistic requirements, translators who are trying to convey the overall message of a text may map the information to the sentences in the target text in a way that differs from how that information was originally dispersed among the source text sentences. The result is that even if the two texts are considered to be equivalent when taken as a whole, the sentences in a translation may not relate to each other in precisely the same way in which the source text sentences do (Bédard 2000).

In order to maximize the “recyclability” of a text, translators working with a TM may choose to structure the sentences in the target text to match those in the source text, and they may choose to avoid using pronouns or other references. According to Heyn (1998:135) and Morgensen (2000:28), the result may be a text that is inherently less coherent or readable, and of a lesser overall quality. Bédard (2000) describes this as a “sentence salad” rather than a text.

The sentence salad effect is exacerbated when the sentences in a TM come from a variety of different texts that have been translated by different translators. Each text and translator will have a different style, and when sentences from each are brought together, the resulting text will be a stylistic hodgepodge. It is highly unlikely that the source text has been created in such a fashion (i.e., by asking a variety of authors to contribute individual sentences), and Bédard questions whether this approach should be used to produce a translation, which is also a text in and of itself.

4. Pilot Study

As previously noted, TMs have only been widely available for a few years, which means that, while there has been lots of speculation about the impact or potential impact of these tools on translation, there have been relatively few empirical investigations. To the best of my knowledge, those studies that have been conducted have focused either primarily on productivity (e.g. Webb 1998) or primarily on quality (e.g. Merkel 1998). I therefore decided to undertake a small pilot study with the aim of investigating the impact of TM use on both translation productivity and quality. This pilot study was carried out in early 2004 using student translators from the School of Translation and Interpretation of the University of Ottawa, Canada, as participants.

The decision to use student translators rather than professional translators was made primarily for reasons of convenience: as a translation professor, I had access to a group of students who were willing to participate in this project. In addition, it seemed logical to conduct an initial pilot study with students in order to work out any methodological problems before asking professional translators to participate in a future study. While there are unquestionably some differences between student and professional translators, given that the latter have more experience than the former, these differences are not so significant as to nullify the results of research conducted using translation students as subjects. For instance, Lörscber (1996:30), who set out to compare the translation processes and strategies used by non-professionals and professionals, makes the following observations:

The first point I would like to make is that in spite of the differences, professional and non-professional translation processes have many features in common. The fact that the categories of my model of analysis, which were developed on the basis of non-professional translations, adequately capture professional translations, too, highly suggests that the two kinds of mental processes are similar, to say the least. From the point of view of the strategies detected, the mental processes of the two kinds of translators did not reveal significant differences.

It is important to note, too, that the non-professional translators who participated in Lörscber’s experiment were not translation students but rather students of English as a foreign language. Therefore, they had not received any translator training per se. Meanwhile, other scholars, such as Tirkkonen-Condit (1990) and Jääskeläinen and Tirkkonen-Condit (1991), have also conducted research aimed at comparing the translation behaviour of professional and non-professional translators. In their experiments, they used first year students to represent the “non-professional translators” and final year students to represent the “professional translators”. Presumably these researchers felt justified in considering that students who were on the verge of graduating from a professional translator training programme – in this case the Savonlinna School of Translation Studies in Finland – could reasonably be equated to professional translators, at least for the purpose of their research.

For the experiment described in the present paper, nine student volunteers participated as translators. The students were all anglophones and were all in the final semester of the honours BA in Translation (French-English) programme. Each of them had previously completed a course in translation technology where they learned how to use the Trados Translator’s Workbench TM system. The students were randomly divided into three groups of three. Students in Group A were asked to translate a text without the help of a TM; however, they were free to use any other resources (e.g., dictionaries, term banks). Students in Group B were asked to translate the same text using a TM that had been previously prepared. Students in Group C were asked to translate the same text using a modified version of the TM – one that had been deliberately seeded with a number of errors.

The aligned source and target texts in the TM consisted of 32 job advertisements for translation positions. These bilingual (French and English) job ads were distributed by the Association of Translators and Interpreters of Ontario (ATIO) between January 2000 and December 2002. Unlike the short ads that appear in most newspapers, these ads are quite detailed and range from 111 to 473 words in length, with an average word length of 283. The text to be translated by the students was a 387-word French job advertisement.

While job advertisements may not be texts that are typically encountered in the localisation industry, the decision to
The use of job ads for this pilot study was based on a number of factors. Firstly, this text type contains a reasonable amount of repetition, as well as a relatively limited vocabulary and syntax. For these reasons, such texts are good candidates for use with TMs. Furthermore, since students were being asked to volunteer their time to participate in this experiment, it was necessary to find a text that would be reasonably short, and yet still self-contained, as well as a text type with which students would be familiar, and which is not so specialized as to require them to do copious amounts of additional research. I felt that I could reasonably expect final-year translation students to produce good quality translations of this type of text.

Two different versions of the TM were produced: TM-O (TM-Original) contained the job advertisements exactly as they originally appeared, while TM-M (TM-Modified) was modified to include a number of errors. To create TM-M, we took TM-O and deliberately seeded it with ten errors of varying types, as shown in Table 4. These errors were introduced into segments that the system would then present to translators as exact or fuzzy matches. Of the ten errors, five were added to exact matches and five to fuzzy matches.

A pre-translation analysis of the text to be translated showed that it contained 36 segments. For seven of these, TM-O contained exact matches and for another six, TM-O contained fuzzy matches. In addition, a term base containing 33 term records was integrated with the TM and was available for consultation. Students translating with the help of TM-O were also permitted to use other resources, such as dictionaries or external term banks, as desired.

All the students were given 40 minutes to translate the text. This is quite a short deadline, but one of the things I wanted to test was whether the pressure to translate quickly led students to use the tools unwisely (e.g. to blindly accept proposals offered by the TM even when they were incorrect or inappropriate to the context in question).

5. Results and Discussion

The results of the pilot study will be discussed below in terms of both productivity and quality for each of the three groups of students.

5.1 Productivity

With regard to productivity, the results of this experiment were not too surprising.

Group A: Two of the students did not finish the translation. One of them completed approximately three-quarters of the text while the other completed approximately five-eighths of the text. These two students said they were reasonably happy with the quality of the work they had completed, but would ideally like more time both to finish and to revise their work. The third student did finish but emphasized that it was only of draft quality and more time would be needed to produce a polished translation.

Group B: All the students in Group B finished the translation. In fact, one of the students finished in 22 minutes and another in 26 minutes, while the third student used the full 40 minutes.

<table>
<thead>
<tr>
<th>Original source text extract</th>
<th>Original translation (in TM-O)</th>
<th>Modified translation (in TM-M)</th>
<th>Type of modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>de façon autonome</td>
<td>autonomously</td>
<td>automatically</td>
<td>incorrect equivalent</td>
</tr>
<tr>
<td>un baccalauréat en traduction plus au moins deux années d’expérience</td>
<td>B.A. in translation plus a minimum of two years translation experience</td>
<td>diploma in translation plus a minimum of twelve years translation experience</td>
<td>incorrect equivalents</td>
</tr>
<tr>
<td>de l’anglais au français</td>
<td>English to French</td>
<td>French to English</td>
<td>reversal of information</td>
</tr>
<tr>
<td>compréhension de base des principes du syndicalisme et de l’équité</td>
<td>a basic understanding of trade union and equity principles</td>
<td>an [...] understanding of trade union and equity principles</td>
<td>omission of information</td>
</tr>
<tr>
<td>aptitude à communiquer verbalement en français et en anglais</td>
<td>the ability to communicate orally in French and English</td>
<td>able to communicate [...] in French and English</td>
<td>change in syntactic structure and omission of information</td>
</tr>
<tr>
<td>effectue de la recherche terminologique</td>
<td>research terminology</td>
<td>effectively research terminology</td>
<td>addition of information</td>
</tr>
<tr>
<td>Prière de faire parvenir votre curriculum vitae</td>
<td>Please submit your resume</td>
<td>Please ensure that your resume and three letters of reference are submitted</td>
<td>change in syntactic structure and addition of information</td>
</tr>
<tr>
<td>Nous adhérons à l’égalité d’accès à l’emploi</td>
<td>An equal opportunity employer</td>
<td>We adhere to egalitarianism for accessing jobs</td>
<td>change in syntactic structure and very literal/awkward phrasing.</td>
</tr>
<tr>
<td>l’orthographe et de la grammaire</td>
<td>spelling and grammar</td>
<td>orthography and syntax</td>
<td>change in register</td>
</tr>
</tbody>
</table>

use job ads for this pilot study was based on a number of factors. Firstly, this text type contains a reasonable amount of repetition, as well as a relatively limited vocabulary and syntax. For these reasons, such texts are good candidates for use with TMs. Furthermore, since students were being asked to volunteer their time to participate in this experiment, it was necessary to find a text that would be reasonably short, and yet still self-contained, as well as a text type with which students would be familiar, and which is not so specialized as to require them to do copious amounts of additional research. I felt that I could reasonably expect final-year translation students to produce good quality translations of this type of text.
40 minutes allotted. All of the students said that they felt happy with the quality of their final translation.

Group C: All the students in Group C finished the translation, and all used the full 40 minutes allotted for the experiment. Two of the three said they were happy with the quality of their final texts, while the third student indicated that it would be desirable to have more time in order to proofread and edit the translation.

5.2 Quality
To evaluate the quality of translations, I used the “official” translation provided by ATIO as a guide. Of course, it is certainly possible to produce different, yet equally valid versions of a text, so I also had to rely on my own judgement and experience as a certified translator and translator trainer. In a future, scaled-up version of this experiment, I would like to solicit the opinions of several independent evaluators; however, for this pilot study, I acted as the sole judge of quality.

Group A: As noted above, two of the students did not complete the translation. One of them was approximately three-quarters finished and the other approximately five-eighths finished. In the case of both of these students, the quality of the work they had completed was quite reasonable. There were no major errors, and no omissions or additions of information. There were several instances of inelegant or awkward turns of phrase (e.g. sentences that were grammatically correct but were not idiomatic or did not flow well). The third student in Group A had completed the translation but noted that it was only a first draft and could not be considered a “final version”. This was evidenced by the fact that, in several places, the student had indicated a range of terms under consideration (e.g. uniformization/standardization, direction/supervision) and had clearly not finished the research that would allow a final decision to be made. In addition, there were a number of sentences that followed the syntax of the French original quite closely and which sounded unidiomatic in English, even though they were grammatically correct. There were, however, no major errors of meaning.

Group B: The students made extensive use of the segments suggested by the TM. In all cases, the exact matches were adopted with no changes. Furthermore, all of the students appear to have edited the fuzzy matches that were presented rather than translating these segments from scratch. This is not to say, however, that the resulting translations were problem-free. Two noticeable shortcomings were apparent. Firstly, although it is frequently claimed that TMs improve consistency, this is not always the case. Sentences culled from different translations – or even from within the same translation but from different contexts – may contain different terminology. Because the texts stored in the TM were likely produced by 32 different translators, there was inconsistency in the terminology that was presented in different matches. For instance, “relecture” appeared in two different segments in the source text, both of which had a match in the TM. In the first match, “relecture” was translated by “proof-reading” (with no hyphen) and in the second by “proofreading” (with no hyphen). Similarly, “de façon autonome” also appeared twice in the source text and was translated in one TM match as “autonomous” and in another as “independent”. In all cases, the students adopted such inconsistencies into their own translations.

A second problem created by the use of TMs was inconsistent style. Because the texts in the TM database had been translated by different translators, they were written in slightly different styles. For example, while most were written in the active voice, at least one contained a high number of passives. Furthermore, when listing duties or qualifications, some ads used verbs (e.g. translate, revise, proofread), while others used nouns (e.g. translation, revision, proofreading). Because the students used matches that originated from different ads, their texts ended up suffering from the “sentence salad” effect – a series of sentences that were inconsistent and non-parallel in terms of their style (e.g. a list of qualifications containing a mixture of verbs and nouns).

Group C: These students were the ones using the TM that had been modified to include a number of errors that were presented to them in both exact and fuzzy matches. In addition to the problems of inconsistent terminology and style that were experienced by the Group B students, the students in Group C also encountered some additional difficulties caused by the errors that were deliberately built into the modified TM. Table 5 summarizes the number of errors that the students incorporated into their translations.

Table 5. Number of TM errors that students incorporated into their own translations

<table>
<thead>
<tr>
<th>Students in Group C</th>
<th>Deliberate errors seeded in exact matches (5)</th>
<th>Deliberate errors seeded in fuzzy matches (5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Student 2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Student 3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>13</td>
<td>4</td>
</tr>
</tbody>
</table>

As indicated in Table 5, only one of the three students in Group C questioned the errors that were presented as exact matches, while the other two students incorporated all of these errors unquestioningly into their own translations. In the case of the fuzzy matches, all of the students recognized and fixed the majority of the errors that were present, although a few did slip through. The probable reason that the students were able to fix the errors in the fuzzy matches more easily is because they know that a fuzzy match will not be perfect and so they expect that they will need to do some editing. This means that they are more critical of a fuzzy match that is presented and they examine it more closely. In the case of the exact matches, the students were often too trusting and they blindly clicked on OK to accept exact matches as they were presented. This blind trust may have been exacerbated by the fact that they were working to an extremely short deadline.
6. Concluding Remarks

Although this pilot study was conducted on a very small scale, it nevertheless revealed some interesting trends. In terms of productivity, it is clear that the use of a TM can indeed help translators to work more quickly. This is supported by the fact that all six of the students using a TM were able to complete their translations within the allotted 40 minutes, whereas the students who did not have access to a TM were unable to complete their translation (2 students) or could only complete a rough draft (1 student). However, being able to translate more quickly is only part of the equation – translators also need to be able to ensure the excellence of their work.

This small experiment seemed to indicate that, when faced with the pressure to translate quickly, translators using TMs may not be critical enough of the proposals offered by the system. The students in Groups B and C tended to accept the suggestions offered by the TM even if these contained errors or did not fit well into the context in question. As previously mentioned, the translations stored in the TM are provided by a human translator and therefore, a prerequisite for obtaining a high quality result from a TM is that the translations stored there by humans must be correct in the first place. If the contents of the TM are of poor quality, then the translators must spend time correcting the translations, and they risk losing some of the time that they may have saved by working with the TM in the first place. Furthermore, it is important to note that even though translations may be accurate when they are initially stored in a TM, they may become inaccurate over time (e.g., if terminology changes) or they may be inappropriate in a given context (e.g., if they contain homonyms or are in an inappropriate register). Consequently, quality control must be treated as an ongoing process, and it is not advisable for translators to reuse previous translations without first verifying that they are indeed correct. This may lead to a conflict between the promise of increased productivity and the need to maintain high quality.

It is easy to see how, when faced with pressures to increase their output, translators may be tempted to work too quickly when using TMs, blindly substituting proposed translations without verifying their accuracy, but this may compromise the quality of the final product. This type of “blind faith” behaviour indicates a definite need for more training in the appropriate use of technology, especially among novice translators, who may not yet have the experience or confidence required to question the suitability of the TM’s proposals. When training translators to use TMs, it is very important to stress that the contents of the TM may not always be correct or suitable in every context. Translators must be reminded that they are still responsible for double-checking the proposals put forward by the system, and for reading and revising the text as a whole (rather than just reading and revising individual sentences) in order to produce a text that is accurate as well as readable. If a translation is produced by recycling individual sentences from a variety of different texts – which may have different terminology, different styles, pronouns or deictics that are unclear without a larger context – the result may be more of a “sentence salad” than a coherent text. This extra text-level revision may mean that the increase in productivity will not be as significant, but taking the time to verify the appropriateness of the TM’s suggestions will help translators to achieve a better balance between productivity and quality.

A final point to note is that it may not be worth including all texts in a TM. If a text has a very unique style or uses in-house terminology that is not common to the field as a whole, including it in a TM could actually be detrimental as it may exacerbate the “sentence salad” phenomenon. In other words, a smaller TM containing well-chosen texts can actually be more useful than a large TM containing a wide range of texts. Translators may get fewer matches overall, but the matches they do get will be of higher quality and will likely need less revision.

Notes

1 Note that even in cases where Internet users do not intend to make an online purchase, they often use the Web to research products that they intend to buy, so manufacturers and retailers would be wise to provide product literature in multiple languages (ABI 2002: 3-5).


4 There are a number of commercially-available TM systems, including Trados Translator’s Workbench, STAR TransIT, Déjà Vu, and SDLX among others.

5 Detailed descriptions of how a TM operates can be found in sources such as Bowker (2002) or Somers (2003). It will be useful, however, to present a brief overview in section 2 for the convenience of readers.

6 A detailed discussion of automatic alignment is beyond the scope of this paper; however, it is worth noting that this is a non-trivial process since the lack of one-for-one correspondence between languages may lead to problems (e.g. a single source text segment may be translated by two target text segments or vice versa, or some information may be omitted from or added to the target text). For a more detailed discussion of alignment, see Bowker (2002).

7 It is important, however, to keep in mind that while using a TM may speed up the actual translation process, translators who wish to use this type of technology may be required to make other types of time investments. For example, they will be required to learn how to use this sophisticated software, and they will need to spend some time pre-processing the texts (e.g., aligning texts, converting to different file formats, stocking the glossary). This means that translators may actually see an initial drop in productivity during this learning phase; however, productivity will hopefully begin to increase as they become more comfortable with the software and pre-processing requirements.
At the time this article was written, O’Brien was working for International Translation and Publishing Ltd. in Ireland.

For example, as observed by Merkel (1998:145), “when two source sentences (or segments) occur in different structural contexts, such as headings and table cells, translators should be more cautious in applying consistent translations.”

Most TM systems come with a module that will quickly compare a new source text against the TM database and calculate how many matches a translator can expect to find.

This advice is generally given to all translators, whether or not they are working with the help of technology. Scholars who emphasize the importance of treating a text as whole include Baker (1992) and Hatim and Mason (1990), among others.

References


Lynne Bowker has a PhD in Language Engineering from UMIST and is currently a professor at the School of Translation and Interpretation at the University of Ottawa in Canada. Her research and teaching focus on translation technology and corpus-based translation studies. She can be contacted at lbowker@uottawa.ca
African Languages and Information and Communication Technology: Localising the Future?

1. Introduction

The dual processes of internationalisation and localisation of information and communication technologies (ICT) are still felt incompletely and unevenly in Africa. Situated on the disfavored side of what is commonly referred to as the “digital divide,” Africa has not benefited directly from internationalisation of ICT as much as it might, largely due to low connectivity and lack of access, but also to the fact that some aspects of internationalisation are not widely available or well understood. Africa also has not yet seen much activity in the matter of localisation for its languages and cultures even though this is vitally important for the future use and utility of ICT in the region.

As a continent with perhaps 2000 languages and various special script requirements, Africa stands to benefit greatly from internationalisation and localisation of ICT, but it lacks the means and strategies to take advantage of them. Moreover, the majority of discussions about “bridging” the digital divide in Africa and how to make ICT work for its people pay little attention to these processes and make only passing mention of the language dimension. This, even though a majority of the population either does not speak or does not master the originally European tongues that both serve as the official languages in their countries and are dominant in computing and the internet.

The future development of Africa, and indeed all world regions, depends increasingly on the appropriation and appropriate application of ICT. These in turn cannot be fully achieved without taking full advantage of internationalisation in order to progressively implement localisation of the technology in order to respond to its cultural and linguistic, as well as socio-economic, needs and realities. Localisation emerges as a key to successful adoption and use of ICT and thus also as an indispensable consideration as Africa looks to the future.

Localisation, of course, has several definitions relating to the adaptation of computer applications and/or the content of computing to the linguistic and cultural realities of a particular country, region, or national community. The sense used here focuses on practical linguistic aspects and addresses both software and content. Nevertheless, localisation is understood in the context of emerging discussions of “ethnocomputing” (Tedre et al 2002) and is captured in Daniel Yacob’s (2004) broad definition: “the transfer of cultural consciousness into a computer system, making the computer a natural extension of the society it serves.”

This paper therefore focuses on the status of African language localisation of ICT and outlines elements of a strategy to facilitate this process. It underlines the importance of two overlapping areas of action – content and tools – and also of prerequisite factors like wider use of Unicode, to successful localisation in, for and by Africa.

II. African Languages and ICT: On the Periphery of the Information Society

A. Importance of the Issue

There are several reasons why African language use in ICT is of interest.

1) So long as a language is spoken and used in other spheres of activity, it is assumed that it is worth providing the opportunity for it to be used in various ways by its speakers in computer and internet technologies. This thinking could be both framed in terms of “linguistic rights” and taken as a practical observation that opportunities for expression and learning are favored by providing means to work with a powerful technology in one’s first language.

2) Such an opportunity becomes a critical concern because although African languages are themselves important vehicles for communication and the creation of knowledge, there is little organized educational activity beyond small-scale programs for adult basic literacy and a limited amount of first language instruction at the primary school level in some countries. ICT in African languages could be important in a range of activities favoring dissemination of information, enhancement of skills, and generation of knowledge.

3) Reliance almost exclusively on English, French, and Portuguese for the transmission of new knowledge (see Enguehard and Mbodj 2003) puts people who are not skilled in these languages, and arguably the entire societies of which they are a part, at a disadvantage.

4) The issue certainly goes further, as it is legitimate to ask what sort of future there is for languages that are not used actively in ICT.

1 ICT is used here mainly in the narrower sense of computers and the internet, what has also been called simply “information technology” (IT) and also the “new” ICTs. In wider usage, ICT can also cover radio, telephones, and other electronic media, the boundaries among which are becoming ever less clear in the wake of rapidly changing technology.

2 The uneven distribution of computers and internet access in favor of certain regions and groups. Various statistics on Africa like low numbers of telephone connections, lack of electrification and high illiteracy that have long characterized the continent are now joined, unsurprisingly, by low indicators for connectivity and access to ICT.

3 Jacques Ngangala Balade Tongamba (2001) sees ICT as presenting an opportunity “without precedent” for Africa, in part because of how it can facilitate use of African languages.

4 Clinton Robinson (1996) cites figures as high as 90 percent in some countries of sub-Saharan Africa who do not speak the official language of their country (English, French or Portuguese).

5 African languages are defined here as the modern languages indigenous to the continent plus major historically indigenized languages (such as Arabic, Malagasy and Afrikaans).
Lack of usable multilingual capacities on computers and the internet in Africa – whether because of old operating systems, lack of necessary fonts and input methods, or system managers insufficiently trained in aspects of multilingual computing – effectively limits many people’s access to full use of the technology. This includes the capacity to display and share text with extended characters or non-Latin scripts – precisely the basic problem that Unicode is intended to resolve – as well as more sophisticated adaptations and uses of the technology, from software localisation to manipulations and transformations of content.

Localisation of the technology therefore involves the entire orientation of the technology with regard to language capacities. In Africa, however, it emerges as both a casualty of language problems (illiteracy, language and education policies that in principle or through incomplete application disfavor African languages) and as part of the solution to those same problems (by opening new possibilities for use of the languages and dealing with multilingual issues).

In the following, the current state of African languages and ICT is briefly surveyed and factors affecting their level of use are examined.

**B. Brief Overview of the Current State of African Languages and ICT**

The relative level of use of African languages in computing and on the internet is hard to gauge but important to attempt to characterize. To begin with, it seems clear that African languages are not yet widely represented in computing applications or internet content. We can deduce this for instance from the very small amount of software localised even for major African languages and the infrequency and character of such web content as one does find in African languages.

This situation obviously arises from the underlying sociolinguistic, language policy, and educational contexts, though in this paper these will not be explored in depth. However, it is worth noting that computers and the internet, like formal educational systems a century earlier, have been introduced and disseminated as more or less monolingual media relying on one or another European language. This is a reflection of both the dominance of the languages inherited from colonization in software and internet content, and the use of these languages by those people in Africa most likely to have access to the technology.

A quick overview of web content, use in e-mail, use in non-internet applications, and localisation of software and web interfaces, helps to elucidate the situation.

**1) The Web**

African languages are represented on the web, but not prominently as media of communication. The actual level of use is emerging as a topic of discussion.

It is easy to get the impression that African language content is rare. To begin with, simple surveys of web content by language that relied on search engines unsurprisingly did not find enough in African languages – even the ones most widely spoken – to rank them as high as some minority European languages with relatively few speakers.

More focused or systematic surveys have come up with more interesting results. For instance, an informal survey done in Tanzania in 2001 as part of a larger report for the Swedish International Development Agency estimated that ten percent of websites with a Tanzanian focus had at least some Swahili content (Miller Esselaar Associates, 2001), but most of the sites did not have majority content in the language. A more extensive study by Diki-Kidiri and Edema (2003) found a significant number of sites that treat African languages in one way or another, but these generally have minimal content in the languages themselves. In effect, a large proportion consists of sites about African languages, including online dictionaries and instructional pages.

On the other hand, Van der Veken and de Schryver (2003), using a different search methodology and statistical extrapolation, suggest that there may actually be more African language web content than we realize.

A useful schema to attempt to understand this evolving situation is that proposed by Ballantyne (2002) for analyzing content in terms of two parameters: “expression,” or whether the content is of local or international origin, and “application,” or whether the audience is local or international. Much web content relating to Africa, even concerning African languages, and whatever the origin, has an external and largely non-African audience, and so would logically tend to use languages understood internationally. Also, much of the web content with intended local application originates from outside of Africa, where production of content in languages other than English, French or Portuguese is not an easy option. What is at issue here is in part the extent to which content originating outside of Africa is adapting to use of African languages, but more importantly the evolution of content of local origin and local application that can naturally and efficiently use those languages.

**2) E-mail and E-mail Lists**

E-mail has long been a significant use of the limited internet connectivity in Africa. By its nature it is harder to track the contents but there is other information that can be used to get an indication of the use of African languages for this purpose. For instance, at one point there were two web-based e-mail services that provided for composition in several African languages: Africast.com and Mailafrica.net (though the latter no longer seems to function). In addition, recent years have seen the setting up of a number of e-mail fora in which much or most of the traffic is in one or another African language. For instance there are several Hausa and Swahili lists in which these, probably the most widely spoken indigenous tongues on the continent, are the primary languages of communication, and Van der Veken and de Schryver (2003) found fora in Hausa, Somali, and Lingala.

---

6 This is a broader definition of access than that commonly used in discussions of the digital divide. However, there are some sources that offer multiple definitions of access. For instance, Telecommunications (2000) discusses “physical access” to ICT infrastructure and applications, and “soft access,” which we define as software and applications which are designed to enable rural African users to utilize ICTs for their own needs and uses once the physical access has been established.” The organization Bridges.org goes further to define twelve dimensions of what it calls “real access,” of which “relevant content” mentions language (see http://www.bridges.org/digitaldivide/realaccess.html).

7 A simple survey of websites by language done in 2000 by Vilaweb, the website of a Barcelona newspaper (Pastore 2000), listed no African languages among the 31 ranked and showed many more pages for languages such as Basque and Slovenian than for any language in Africa surveyed three years later by Diki-Kidiri and Edema (2003). A follow-up to the Vilaweb survey which ranked the top 48 languages on the web found Afrikaans 42nd after the abovementioned languages, and Swahili last following, among others, Frisian and Faeroese (Mas 2003).

8 There was even a “web-page by e-mail” service hosted for several years by Kabissa.org, in recognition of the fact that many people in Africa could not access the web but did have limited e-mail access.

9 These include several that use the free Yahoo groups service: “Kiswahili” http://groups.yahoo.com/group/Kiswahili/; and for Hausa language: “Finafinan_Hausa” groups.yahoo.com/group/Finafinan_Hausau “HausaDaHausawa” groups.yahoo.com/group/hausadahausawa,” “Marubuta” groups.yahoo.com/group/Marubuta,” “Matasa” groups.yahoo.com/group/Matasa/. The scholarly-oriented H-Swahili list also has traffic in the Swahili language: http://www.h-net.msu.edu/~swahili
3) Non-Internet use of African languages in computing

It is harder still to attempt to quantify the degree to which African languages are used in the content of computer applications in Africa, for instance on word processors for the production of printed documents. Among users in this category there have been specialists in African languages and linguists in Africa and beyond. Certainly the publication of books and news in African languages is computerized using specialized publication software, but use on public, office and personal computers is less visible.

As for popular usage, a glimpse of African language use on computers in a Senegalese telecenter is given in a brief article (Elder 2002) that mentions use of Pulaar and Wolof. Also in Senegal, a local non-governmental organization, ANAFA, has been doing computer training (including basic literacy) in national languages. Beyond such anecdotal evidence however, there are apparently no surveys of such non-internet use.

4) Software and web-interface localisation

Localisation of software and web-interfaces for African languages is getting increasing attention. There have been efforts for localising software on smaller scales for several years, but these are becoming more numerous and the level of activity is increasing.

The recent agreements concluded by Microsoft Corporation to localise for Swahili (Anyanzwa 2004), Wolof (APS 2004), and Hausa, Yoruba, and Igbo (This Day 2004) indicate the importance that the company is attaching to the issue. The announcement early last year (Microsoft Corporation 2004) concerning its increased work on localising its software, including for Africa, follows on preparation that apparently goes back some years.

Localisation of open-source software is also emerging as an important factor. A localisation project for South African languages, Translate.org.za, has received a fair amount of attention, announcing completion of software in Zulu, Sepedi, and Afrikaans last year. In late 2004 another open source localisation project, in Uganda, released a web browser in Luganda (Otter 2004), and a Tanzanian-based project, Kilinux, completed a Swahili-language word processor. Other projects are in the works though information on them is sometimes not readily available.

There are also some African projects that have produced software for composition in African languages but without language localised commands, including several based in or focusing on Nigeria - Kinyi, Afar, ALT-1, and Paradigm. As for web-interfaces, the popular search engine Google has had a program for localised versions that already has several African language versions translated by volunteers. Another example is a “V-webmail” e-mail interface which was recently localised for Swahili. There may be more of this sort of language localisation going on than is apparent, but one concern with easier localisation such as that offered by Google is the quality of the translations done by volunteers.

C. Factors Affecting the Level of Use of African Languages in ICT

Despite the examples cited in the previous section, African language use in ICT appears to be relatively marginal in Africa. Why is that? First of all, the factors that define the digital divide also tend to minimize the potential for African language use in ICT. Connectivity is centered on cities and towns where official languages – the same languages that are dominant on the internet – may be more widely spoken. In addition, only people with means and education, who are also more likely to have facility in use of the official languages, can access computers and internet connections. The digital divide therefore is arguably more localised than bridged, being replicated on national and local levels along the lines of older and deeper social, economic, and linguistic divides. In effect there are a number of interrelated factors that favor languages that are not associated with wealth and power, even before one considers possible issues specific to particular languages.

The other reasons why African languages are not more used in ICT can be grouped into two categories: motivation (the will to use them) and structural factors relating to the languages that affect access to and use of ICT (the means to do so).

1) Motivation

With regard to the motivation, it might be said that while those people who have access to computers in Africa tend to be people who are educated in and socialised to some degree to use the official languages and thus less likely to actively seek to use their first languages in ICT, those people who use their first languages but not the official language tend not to be in a position to do much in this area even if they wanted to. Of course the issues surrounding use of the vernacular in Africa are complicated by factors such as status and attitudes towards what is indigenous (including language) vis-à-vis languages that are seen as providing more economic opportunity. It becomes easy then for all to assume that ICT is for European languages. This is all the more so since for many internet users, the medium is primarily a way to access information from other parts of the world.

Another factor in the case of web content has to do with the intended audience(s) to which web sites are addressed. Website designers who are concerned primarily with addressing content...
Language Localisation of ICT

There are several structural factors limiting African language use. Some of these relate to standardization of orthography, which in some cases is subject to change or individual experimentation, and in more than a few cases varies for the same language across borders. A significant number of less widely spoken languages apparently do not have any established orthographies.

Another factor is that of the special characters or non-Latin scripts used in many orthographies that required specialized 8-bit fonts but now can use Unicode fonts. However, there is still some debate on the provisions in the Unicode standard for certain diacritical characters used in some languages, which currently pose some inconveniences (see Tassé 2003). On the whole, though, the problem is that the use of Unicode is still not widely understood among computer technicians and systems administrators on the continent.

This in turn relates to a lack of intersection of language policies and ICT policies in most African countries. In fact, it appears that there is little collaboration between linguists and ICT technicians in Africa outside of programs like those of RIFAL (Réseau international francophone d’aménagement linguistique). Similarly, in development agencies there is generally a lack of knowledge about African languages and linguistics or about basic technical options to facilitate computing in multiple languages (especially in the case where characters or scripts beyond standard ASCII are used). The opinion of the individual with DFI mentioned above concerning the prospects for African languages in ICT is merely a more extreme example of lack of knowledge of the realities and possibilities.

Another structural factor that is as essential as it is obvious, relates to lack of resources to advance work in these categories, even where there is the will and know-how to implement multilingual ICT projects in Africa. Often though, the know-how is insufficient especially outside of technical circles that are most often based uniquely in the capital city, so plans for training to build skills of local level technicians is necessary.

III. Elements of a Strategy for African Language Localisation of ICT

A strategy to favor African language use in ICT needs to concentrate on two main concerns: favoring dissemination of content and providing tools. In other words, any concentration on what the technology can do with African languages to meet people’s needs and aspirations should focus on what it can deliver and how it can best facilitate more production in these languages. Also, these two concerns are essential in expanding access to ICT. These will be discussed below.

In addition, and as a prerequisite (or corequisite), structural factors including those mentioned in the previous section, that together affect the “environment” for localisation, need to be addressed in order to facilitate content production and providing tools. Some of these are beyond the scope of this paper but bear reiterating: basic literacy (which also can be understood as enhancing user profiles), language and education policies (what priorities do the governments have in terms of learning in and other use of indigenous tongues, and how do these priorities mesh with ICT policy), and availability, quality and cost of connections outside of the capital cities (some innovative use of the technology for rural Africa might require significant bandwidth). The others will be outlined below before discussing content and tools.

These structural factors include: supporting existing efforts and improving coordination and collaboration among people active in the field; standardization of orthographies and spelling conventions; and dissemination and adoption of Unicode continentwide.

A. Structural or “Environmental” Factors

A trio of factors needs explicit attention as prerequisites or at least corequisites of successful efforts to develop content and tools:

1) Supporting ongoing efforts and improving coordination and collaboration among the actors in the field (software developers, content managers, etc.)

2) Standardizing orthographies, rules of transcription, and locales

23 This and similar comments heard by the author in Africa and also regarding indigenous languages in the U.S. echo Keniston’s (1999) observations concerning India: “It can be argued that, given the fusion of language, wealth and power in India, there is simply no market (and perhaps no need) for software in any language other than English. Asked about localisation to Indian languages, international software firms sometimes reply, ‘But everyone speaks English in India,’ by which of course they mean that the present market consists of people who speak English.”

24 Such argumentation in the end resembles the vicious circle of rationalization that hobbles foreign assistance to literacy efforts: it is sometimes argued on the one hand that printing materials in African languages is pointless since few people know how to read these languages, and on the other hand that it makes little sense to conduct literacy training in these languages since there is so little to read in them. The author has encountered such arguments in Niger.

25 In Niger, for instance, the most recent revision of orthographies dates to 1999. In the case of the Igbo language of southeastern Nigeria, a recent dictionary (Echueo 1998) departed from established practice by substituting a diersis for the subdot on vowels and /č/ for /č/ - a decision that aroused some controversy.

26 Significant effort has gone into trying to harmonize transcriptions of cross-border languages, including international expert meetings facilitated by UNESCO several decades ago (see http://www.bisharat.net/Documents/). Nevertheless there are frequently small differences.

27 An overview of African orthographies and ICT usage is surveyed by the author elsewhere (Osborn 2001).
3) Adopting and adapting Unicode continentwide, especially for those countries whose languages use non-Latin scripts or extended characters in the Latin alphabet. These factors are interrelated and together they can create a facilitating environment for localisation.

1) Supporting existing efforts
Support for efforts to localise ICT in African languages needs to be considered broadly. Not only are means needed, but also information, networking and collaboration that goes beyond narrow technical issues. Special attention is needed to involve linguists, policymakers, and educators.

Support for efforts to develop mono- or multilingual web content in African languages first needs to take account of who is developing the content for whom, perhaps using Ballantyne’s (2002) schema mentioned above. The support itself can take various forms such as means to finance or encourage such work in Africa, and ways to facilitate communication among people doing it wherever they are.

A workshop on African languages and the internet at the 2002 African preparatory conference in Bamako for the World Summit on the Information Society (WSIS), for instance, proposed establishment of a “Highway of African Multilingual Information” (HAMI) fund to support creation and maintenance of African language web content. Although this proposal does not appear to have been acted on, the idea of donor support for African language content is one that should be seriously considered in any long-term ICT strategy on the continent. The mechanics of such a program could range from something as simple as annual prizes for websites developed in Africa with African language content, to something more complicated such as training, or other approaches.

Support for software localisation in Africa similarly would benefit from material and technical support. At the time of writing, a workshop for African localisation developers is in the planning stages.

Any effort to support African language localisation also needs ways to facilitate ongoing communication among people working on separate projects, including Africans and others in Africa, Africans in the diaspora,29 and others abroad who are motivated to help. There are already several electronic fora for discussion of issues related to African language in ICT that seem to have demonstrated the potential of this medium for fostering exchange of information and even collaboration on small projects. Although some questions about how the standard meets some character needs, the main problem with Unicode is that despite the fact that it is being adopted as the industry standard, it is not yet widely understood by technicians in Africa. One recent example is a telecenter project in Mali where the director indicated that telecenter managers did not know much about Unicode, and that the font they had for national language use is a legacy 8-bit font (ironically one of the same legacy fonts used for documents that the RIFAL project is assisting Malian authorities to convert to Unicode fonts).

Wider training in and discussion of use of the Unicode standard therefore seems to be needed, mainly in those countries (most of them in West and Central Africa) whose languages are written with non-Latin or extended characters. However, since it is intended as a universal standard, and in the interests of African unity, Unicode’s use should be promoted throughout the continent.

B. Content
Content here is considered broadly, including web content and also e-mail and traffic on discussion fora. It is one of the two main strategic concerns of localisation in Africa.

Beyond supporting existing web content creation efforts there is a need to consider other creative approaches to increasing the amount of material on the web that is accessible in African languages. These could include for instance:

1) Putting extant African language texts on the web
2) Increasing the amount of current high-quality Africa-related web content that is translated into appropriate African languages
3) Facilitating use of African languages on internet fora
4) Exploring the potential of audio content

Creation of web content directed to Africa in its diverse languages is a way to make the technology more relevant to the people of the continent. Such creation can be accomplished in several ways, and it is instructive to review these and how ways can be found to speed up the process.

First of all, development of original text-based web content from scratch requires much time and resources. Furthermore, one cannot develop significant amounts of content in African

28 See http://www.bisharat.net/Documents/Bamako2002-workshop.htm. Reference to this was also made at a later WSIS preparatory meeting in Accra in February 2005 (Diakité 2005).

29 The African diaspora’s role in African language web content production seems to be a significant if overlooked factor (this author has touched on it elsewhere – Osborn 2004). Use of internet among diaspora communities in other ways is already a noted factor (Ajibewa and Akinrinade 2003).


31 A series of expert meetings sponsored by UNESCO, of which one in Bamako in 1966 is considered seminal, dealt with these issues and have had an important impact on the transcriptions we see today. Documents from many of these meetings are available at http://www.bisharat.net/Documents. A South African based organization, the Center for Advanced Studies of African Societies (CASAS), is currently working on this issue.
languages very quickly, as there is a lack of available human and monetary resources to apply to the task and in some cases the orthographies are not settled or require specialized scripts. In any event, the dynamic here is by its nature slow: the proportions of the languages on the internet change, but gradually. Some expect that these proportions will continue to approach an approximation of the current percentages of speakers of languages in the world, but that assumes relatively comparable written traditions and available resources – conditions that don’t hold for much of Africa.

1) Putting extant African language texts on the web
One quick way of increasing quality African language content is to put published and other extant texts on the web. This could include texts published in African languages (often with parallel text in a European language) and monographs with African language content. Such literature, currently accessible only to limited audiences would thus be accessible to a new generation of readers. In addition, a range of other, unpublished materials could also be made available on the web.

In addition to obvious cultural importance, the presence of such material online could encourage development of other original content in African languages and help set a high standard for presentation in them.

2) Translating Africa-related content
The content on the web is vast and continually changing, so no one would propose translating it or any major part thereof into any language, however important. However, some content relevant to culture, development, education, health, and agricultural and natural resource extension in Africa might be very appropriate and useful to have translated into various African languages. The focus could be on relatively static content on basic themes and be approached as an investment, much like publishing books.

This process would be facilitated by advances in machine translation for African languages (see below, III.C.4).

3) Facilitating use of African languages on internet fora
The existence of e-mail fora in which African languages are used has been noted above (II.B.2). In cases where the languages in their official orthography use characters not supported by existing web-based groups or distribution list archives, efforts should be made to use existing technology to make them Unicode aware. Web-based groups could also facilitate input of extended characters, for instance, by the use of click-on buttons or pop-up keyboards.

But more than the technical solutions, the use of the languages in various kinds of announcements and mailings by projects and people with the means to do so will demonstrate the potential for African language use in this medium, and develop models for their use by others.

As the growing movement to publish weblogs becomes more common in Africa, software for these should also be Unicode compliant to facilitate diverse language use.

4) Exploring the potential of audio technology
African cultures are often referred to as “oral cultures”, and even where the written forms of its languages are being used, the importance of speech and orality is still high. It seems therefore worthwhile to explore the ways in which the audio capacities of computers and the internet can be better exploited to create relevant content.

Improvements in audio technology can in theory allow a vast quantity of spoken content in any language to be quickly added to or communicated on the internet. Audio files will never replace text (part of the problem is their size), but they make it possible to create new forms of content with various combinations of text, image and audio (based, if so desired, mainly on the audio). Audio e-mail, for instance, is a marginal technology in the Northern countries but might be very interesting to develop for users in multilingual societies of Africa and other parts of the global South.

In addition, text-to-speech and speech-to-text technologies could also be important. Could localising ICT in Africa open up new paths for evolution of oral culture as well as for developing written culture?

C. Tools
Tools that facilitate the use of African languages in ICT and the production of material such as web content in them is a second pillar of a strategy to enhance use of these languages. Several specific elements are important, including fonts, keyboard layouts, software localisation, and machine translation. All of these rely on standardized orthographies and benefit from use of Unicode.

1) Fonts
Although many African languages do not have special script or character requirements and thus can be typed in a range of readily available fonts, many others do require fonts with modified Latin letters (extended characters) or non-Latin alphabets. There are not many such fonts and so a basic unmet need is for more quality Unicode fonts to facilitate use of these languages. This, of course, is both to be able to view text on the web and to be able to compose text for diverse purposes. Various means to promote such font development need to be examined – for instance if it will work best on proprietary or open-source levels – and implemented.

It would be ideal for Africa if the major Latin fonts used internationally were to include all Latin character ranges and if other fonts for Africa could include, as a matter of course, all Latin ranges plus all Arabic ranges, Ethiopic/Ge’ez, Tifinagh, N’ko, and yet to be encoded scripts like Vai.

2) Keyboard layouts
Along with fonts there is a need both for means to input the necessary characters and for standardization of layouts. When extended Latin characters or diacritics (sometimes used to mark tone or to indicate alternate sounds) are needed to compose text in a language, keyboard layouts can be designed to facilitate input. This can be done through creation of specialized keyboard layouts for use with existing software, or as part of software localisation.

Interfaces for input of special characters can be done in a number of ways such as using programs like Tavultesoft’s Keyman program, Microsoft’s keyboard layout utility, or simply by assigning keys within a wordprocessor program. These are not particularly hard and in fact there is an increasing number of these available for various languages and countries or regions.

---

32 For example a series of books in the “Classiques africains” collection published over the last three or four decades has numerous titles with parallel French and African language text.

33 Three main approaches include: key combinations (with the Alt or Ctrl keys), deadkeys (where one key typed before another yields a certain character or diacritic after striking the second key), or substitution (simply reassigning a key to another character).

34 See for instance the Tavultesoft site http://www.tavultesoft.com or the keyboard projects links at http://www.bisharat.net/A12N/Projects.
Non-Latin scripts such as the Ethiopic/Ge’ez syllabary used for Amharic and Tigrinya may pose a challenge. In some cases an alternative approach using graphics tablets with keyboard assignment or even handwritting recognition may be useful. Indeed, the potential to use traditional keyboards in tandem with graphics tablets is an option that should be explored in contexts where multiple scripts are used.

An important issue, however, is the standardization of layouts for either individual languages or groups of languages in a country or countries. The object is to provide users with predictable input systems and software localisers with information to guide their work. This stops short of discussing physical keyboards designed to better accommodate multilingual needs in Africa, but these are a logical extension of discussion about (virtual) keyboard layouts.

3) Software and web-interface localisation

As indicated above (II.B.4), localisation of software in African languages by open source and proprietary projects is beginning to diversify as it evolves. Proprietary firms – both multinational (notably Microsoft) and local – are in the field as is a growing open source software movement. In addition, one notes what could be characterized as two different levels of software localisation: full translation of the user interface and software that still relies on a language of wider communication for commands, etc., but facilitates use of specific sets of African languages through making available appropriate fonts and dictionaries.

These efforts should be encouraged and collaboration among the local ones facilitated (per III.A.1, above). One notes for instance that the Translate.org.za project in South Africa has offered assistance to similar efforts in other countries – this kind of mutual assistance can go a long way to building capacity in various parts of the continent to carry out localisation projects.

In the long run, such software localisation provides tools that can be readily used for web content development and may also have a very positive impact on a range of other production in the languages concerned. One might expect that localised software would contribute to a boon in local content.

Beyond promoting collaboration among localisation developers, there are needs to develop continental strategies for training, prioritize types of software, and do outreach in countries where there is no localisation. For each language, moreover, there is also the need to involve appropriate language experts so that the terminology and orthographies conform to operant standards.

4) Machine Translation

Although the technology of computer translation is in some ways still rough, it is being steadily improved. Beyond figuring in any strategy to increase the quantity of materials in African languages available on the web as discussed above, this technology can also be a key part of efforts to develop materials in African languages for publication in print. The object would be the creation of software for translation between the principal official languages (English, French, Portuguese) and African languages, as well as software for translation among the dialects of various African languages.

In any event, the ability to read any text on the internet in the mother tongue (even if the translation is not perfect) could have immense implications, as would the potential for speeding translation of reference and other materials into less widely spoken languages.

As with software localisation, there is a need to plan for training, but beyond that also the opportunity to consider the development of the specialty of machine translation in Africa (including long-term development of a professional association under the International Association for Machine Translation on a par with those in North America, Europe and Asia).

IV. Conclusion

African language localisation is an essential part of any effort to address the digital divide on the continent and use ICT to meet the development needs of its population. This fact is now beginning to get more attention, but there is a need to formulate and elaborate plans to support aspects of localisation. In the current situation of African languages and ICT one sees some useful elements to build on, including of course Unicode, as well as some challenges. At this time, a strategy for localisation of ICT in Africa should focus on facilitating the creation of web content and the development of software tools, and such a strategy should address several elements as outlined above. In the longer run there are other issues to be addressed and probably new dynamics that will come into play. Indeed, by actively taking steps now, the environment for future localisation will be enhanced.

Bibliography


Elder, Laurent. 2002 “What a Difference IT Makes ...” (an Acacia project site visit) http://web.idrc.ca/en/ev-9481-201-1-D0_TOPIC.html


Donald Z. Osborn is the founder and director of Bisharat, Ltd. With a background in environment, agricultural development, and African languages, he has developed an expertise in African language computing and localisation with an eye towards applications for sustainable development. He can be reached at dz0@bisharat.net


Donald Z. Osborn is the founder and director of Bisharat, Ltd. With a background in environment, agricultural development, and African languages, he has developed an expertise in African language computing and localisation with an eye towards applications for sustainable development. He can be reached at dz0@bisharat.net
Conceptualising the future of translation with localisation

Minako O’Hagan, School of Applied Language and Intercultural Studies, Dublin City University

The widespread use of electronic communications modes for international exchanges is making localisation relevant to the emerging need for new language support. Indeed, the recent demand for enabling Web sites in multilingual versions has established localisation as integral to the modern translation business. As a consequence, conventional translation is being increasingly replaced by what the author terms teletranslation as the means to enable translation-mediated communication (TMC) in electronic modes. This paper explores how elements of localisation practice are providing the basic building blocks of teletranslation. The author therefore maintains that localisation provides insights into the future of translation.

1. Introduction

The birth of localisation coincided with the opening of international markets for the computer industry that began to emerge in the 1980's (Esselink, 2000). Although the single largest component of localisation still is translation, the process of localisation as a whole diverges greatly from conventional translation as the latter is primarily based on print media which is the hallmark of the industrial society. Software localisation involves a number of linguistic and cultural adjustments, concerning not only documentation in print, but software itself. The task therefore is a combination of translation in its conventional sense and software engineering. Many early localisation companies sprung up from the software engineering sector rather than conventional translation operators which found the engineering dimension beyond their scope. This origin has meant that localisation tended to create its own path, more akin to software engineering and separate from conventional translation. However, the 1990's saw a sea change in this notion as the localisation industry firmly established itself to be a significant segment of the modern translation business. The clearest contribution was localisation of the World Wide Web (Web).

In the advent of the Internet and its requirements to enable multilingual environments, new modes of language support emerged. One was localisation of Web sites, enabling the user to view and navigate the content of a given Web site in his or her language. Sophisticated Web sites are now offered in multilingual versions and updated simultaneously across all languages. Enabling and maintaining Web sites in a number of languages was almost immediately established as a localisation task rather than that of conventional translation because this involved software engineering in addition to translation. The localisation industry established best practices and led the translation industry into this new era of multilingual Internet. Web localisation cemented the position of localisation as being indispensable for meeting the emerging need for language support on electronic communications platforms.

Needless to say, however, not all Web sites are localised into all languages and this allowed another popular language support to develop on the Internet in the form of online machine translation (MT). Today many portals and search engines integrate MT as part of their basic utilities to enable the user to obtain the required information in his or her chosen language. The rationale for employing MT was clearly linked to the need for (near) real-time translation, seamless integration of language support into the user’s online environment and also cost consideration in line with the ‘free information’ culture still prevalent on the Internet. To this end, MT services on the Internet are mostly provided free of charge. MT-based language support is well suited to meeting the needs that arise from the Web for low cost real-time information ‘jisting’.

As these two examples illustrate, the user environment in which language support is required is changing. Conventional translation, characterised as an asynchronous process catering to print-based text for physical distribution, is becoming less and less compatible with the rapidly expanding ICT-based communications infrastructure. It seems reasonable then to assume that future translation will be increasingly sought in the context of electronic modes of communications based on digital technology. The author calls the emerging language support teletranslation (O’Hagan, 1996; O’Hagan & Ashworth, 2002) to highlight the change from conventional translation. This change will not mean that conventional translation will disappear, but rather, new modes of language support will subsume the function of the former. The term teletranslation is used to mean language support provided to enable interlingual communications in electronic modes whereas the term conventional translation refers to language support in conventional non-electronic modes such as translation of text to be used for offline print media.

2. New Contexts of Translation-mediated Communication (TMC)

2.1 Definition of TMC

In its broadest sense the term Translation-mediated Communication (TMC) can encompass any interlingual exchange enabled by the assistance of a translator or an interpreter. For example, a translation of foreign literature, a subtitled foreign film and a meeting assisted by an interpreter can all be regarded as TMC. The word Translation in Translation-mediated Communication includes interpretation as well as translation. Further, in the context of this paper, the main focus of TMC, as in O’Hagan & Ashworth (2002), is interlingual exchanges occurring in electronic modes such as CMC (Computer-mediated Communication) rather than the conventional modes based on physical print media or face-to-face interactions. The term TMC in this paper therefore takes the narrow meaning of translator- or interpreter-assisted electronic communications. On the basis of Shannon’s (1949) communication model, TMC can be examined in terms of the Sender of the Message in the Source Language (SL) and its Receiver in the Target Language (TL) communicating via Translator (either translator or interpreter) who converts the Message from SL to TL. In particular, the TMC framework focuses on the role of technology by analysing its impact on the Sender, the Receiver and the Message itself in a qualitative man-
ner, thus different from Shannon’s model, which is focused on the transmission function of telecommunications in a quantitative sense. The set of terminology with capitalised first letters (Sender, Receiver, Message and Translator) are used throughout this paper whenever they are referred to in the context of TMC. Nida and Taber (1969) highlighted the role played by the translator on the basis of the Shannon’s model and this is relevant to illustrate TMC.

The fundamental function of the translator as the Receiver of the Message in SL and the Sender in TL remains the same between conventional translation and teletranslation, but in the way in which the Message is transmitted, stored and processed show marked differences. The following section takes the case of localised Web sites and examines it in the framework of TMC.

2.2 Localised Web sites as TMC

In the late 1990’s multilingual Web sites started to emerge in recognition of the nature of this communication platform being immediately global. This created a new domain of language support which requires localisation of a Web site within given locales. In order to highlight the difference from print-based text subject to conventional translation, the following diagram shows the typical lifecycle of Web content also in relation to the Sender and the Receiver of the Message.

The Web allows its users to disseminate and assimilate information in an interactive manner enabled by hypertext. The lifecycle of content for a Website starts with authoring, which may range from entirely text-based to incorporating audio, moving images and other non-textual elements. Once the site is published by the Sender, it goes global unless access is limited such as on an intranet. The Receiver accesses the site by some kind of IT device which today includes mobile phones and PDAs (Portable Digital Assistant) in addition to desktop computer terminals. Once the Receiver comprehends the content, he or she may act upon the received information. For example, the Receiver may decide to place a bid on an item in an auction site. The Sender of the Web site is normally able to gather various user information automatically and may use such feedback to update the site. The cycle continues in this manner. The localisation process is typically applied between the authoring and the publishing stages although an application of the internationalisation process will, in effect, advance the overall localisation planning to prior to authoring. Internationalisation is further elaborated in the next section. Because the whole lifecycle is embedded in the digital environment, all aspects of information storing, processing and transmitting of Web content are conducted by computer. Given the frequent updates required of the information, ongoing content management is a significant dimension of maintaining localised Web sites. This is similar to the continuous cycle of software versioning, but is new to conventional translation which generally deals with text not requiring such frequent updates. In the TMC framework, a localised Web site can be characterised as follows:

The Sender in SL: Originator of the Message.

1. The Sender is increasingly aware that once the site goes online, anybody can access it (unless access is deliberately restricted) and the circulation tends to be wider than that for offline print-based translations.

2. Web authoring has become a specialised technical task as it involves understanding the characteristics of the Web. For example, readability of the content as well as its aesthetics on screen rather than paper need to be taken into consideration. Furthermore, the Receiver environments such as operating systems, browsers, character encoding, etc. also become relevant factors to consider.

3. The application of internationalisation may affect the Sender to allow for subsequent translatability of the content.

The Receiver in TL: Unspecified recipient of the Web site.

1. Localised Web sites will be exposed to a wide range of TL readership and accessed via an increasing range of devices, including mobile phones.

2. Interactivity based on hypertext allows the Receiver to arrive at different parts of the text, not necessarily in the order in which it is originally written.

3. The Receiver will soon discover if the given site is user-friendly in terms of functionality as well as linguistic and cultural dimensions. A localised Web site facilitates the Receiver to understand the content, but depending on the extent of localisation, not all information or functionality may be available in TL. Some Web sites are only partially localised and this may prompt the Receiver to apply ad hoc language support such as online MT.

The Message: Web content

1. In addition to text, Web contents often include non-textual components which may require adapting to the Target culture.

2. Web content needs to allow for the fact that information is often scanned quickly on screen by the Receiver, at least in the first instance to gather relevant information.

3. The increasing use of mobile devices to access the Web has meant that the content needs to be further tailored to smaller screens of such devices.

4. Web content is normally updated at much more frequent intervals than that of offline print media.

The Translator: Web localiser

1. The task of converting a Web site from SL to TL involves translation of text in a conventional sense and software engineering.

2. The Translator needs to be aware that the Web content is normally exposed to much wider TL readership than offline text, thus the content has to sound and look natural to a wide range of TL native speakers.

3. It is essential for the Translator to understand the characteristics of the Web as described earlier.

4. The Translator will have to be familiar with content management requirements and be required to use certain translation tools such as Translation Memory to facilitate efficient updating of the content.
A number of differences from conventional translation are obvious. The impact of the new mode of communication on the overall translation process is clear in response to the changes in the characteristics of the Message, how the Sender has to present the Message and how it is consumed by the Receiver. The overall process involved in localising a Web site is akin to software localisation in that the task is possible only if supported by computer to fit into the digital lifecycle of Web content. For example, unlike conventional translation, quality control involves computer-based testing in terms of the user environment as in localisation of software. The use of certain computer-based tools is also essential — particularly in view of the ongoing maintenance of the content, which results in frequent changes for updates. In what follows, an emerging example of teletranslation is explored.

3. Emerging teletranslation practice: DVD localisation

The localisation process has evolved in response to the need for language support in new technological environments such as computer software and the Internet. With the widespread use of the Internet, localisation of Web sites became the fastest growing area within the translation sector in the late 1990s (Lockwood, 1999). Similarly, the growing use of DVD for audiovisual content may see this medium becoming the next big wave of localisation. DVDs offer enhanced storage and information processing capacities. For example, as compared with CD-ROM which has a storage capacity around 700MB, a single DVD has 4.7 to 17 GB and is able to embed subtitles in up to 32 languages or up to 4 dubbed versions (Karamitroglou, 1999) with both subtitles and dubbed versions on the same disk. Furthermore, DVDs allow interactive features such as scene-based searches, various processing possibilities of selected scenes and user options to show/hide or select subtitles. The capacity of DVDs to support multilingual speech and text will make them an ideal platform for audiovisual content distribution in the global market. The implications of this could be that certain localisation practices such as simultaneous shipping (simship) may become applicable to DVD releases in the same way as for popular computer software and Web sites. This will likely mean that screen translation becomes subject to certain standard localisation processes such as project management, workflow and quality control procedures. Furthermore, the subtitling and dubbing processes themselves may be affected. Given the new interactive features as well as processing and storage capacities pertinent to DVDs, a new mode of screen translation could develop. For example, today’s subtitles as a linear text stuck at the bottom of the screen could change to incorporate hypertext and multimodal content (the concept of making the content, which results in frequent changes for updates. In what follows, an emerging example of teletranslation is explored.

4. Significant dimension of localisation: Internationalisation

4.1 Today’s internationalisation

The author maintains that localisation provides a theoretical basis for emerging language support. One particular dimension which seems to have a far-reaching impact is the concept of making allowance for localisability and translatability when developing the source content in SL. This is the process called internationalisation (see Esselink, 2000; Kano, 1995) increasingly applied to products which are subject to subsequent localisation. Confusion over different definitions of the terms as regards to localisation, internationalisation and globalisation are often pointed out (Esselink, 2000). For the purpose of this paper, the author adopts the definition by Cadieux and Esselink (2002) who suggest: Globalisation = Internationalisation + N x Localisation. This formula shows that globalisation of a product or service involves internationalisation and localisation into a given number of locales. Both internationalisation and localisation therefore can be seen as key steps to achieve globalisation. In particular, internationalisation means preparatory tasks for subsequent localisation and therefore can be understood as localisation-enablement (Cadieux & Esselink, 2002). As such this approach makes a stark contrast with conventional translation which has typically taken the SL text as a given and thus applied to already completed SL text. In fact, one of the major characteristics of conventional translation has been the constraint imposed by SL text in relation to TL text production. By comparison, the internationalisation process aims to deal with foresee-able localisation and translation difficulties in advance at the inception of SL content. The internationalisation process applied to software products involves technical adjustments to externalise all translatable components (Esselink, 2000). This means that internationalised products are designed and developed with such necessary modifications in mind as different character encoding systems (single byte vs. multiple byte characters), different lengths of TL text in relation to those of SL text, different conventions for expressing time, date and any other culturally specific elements in the Receiver context. Internationalisation of e-commerce Web sites may involve adjustments of non-textual elements, ranging from the design of the page, appropriateness of certain icons and images, to payment methods most suitable in the target culture in addition to consideration in terms of user inputs such as different currencies, digits for telephone numbers or the need or absence of postal zip code, etc.

Some may see internationalisation in association with earlier attempts at the use of controlled language applied to the source text. Controlled language is primarily intended to make the Message more machine-friendly by eliminating in advance those elements which are known to be problematic when the Message is destined for computer-based translation. Similarly, the concept of pre-editing of text for the use of MT is not new. However, the most extensive application of internationalisation is a novel approach in that the Message is controlled in its inception in terms of its techni-
cal, linguistic and cultural dimensions in relation to its Receiver.

The internationalisation process can thus be seen as a comprehen-
sive effort to make the Message amenable to the subsequent human-
and machine-based translation processes and the use of controlled
language can be regarded as a subset of the same attempt. As such,
it is a clear contrast to the traditional treatment of translation as an
isolated downstream activity where the Sender in SL has typically
no regard for the Receiver in TL when creating the Message.

It is easy to demonstrate the advantage of internationalisation in
the context of TMC with media such as computer software and the
Web. For example, by recognising the fact that a software product
originally produced in the USA will be marketed in China, Japan
and Korea, technical allowances can be made to accommodate dou-
ble-byte character sets required by these Asian languages as well as
other culture-specific aspects in terms of the overall design of the
software. If such consideration is not given at the beginning of the
software development cycle and is not technically inbuilt, expensive
re-development is likely to be necessary. The question of how the
proactive approach such as internationalisation will impact on the
overall TMC will be relevant in exploring further implications of
internationalisation for emerging language support.

4.2 Future implications

Localisation processes have evolved under highly competitive com-
mercial conditions and therefore cost-efficiency is a key factor. This
focus is manifested in such tools as Translation Memory and work-
flow programs to shorten the time of localisation while maintaining
the quality of the output. The need for internationalisation also
arose to conduct a localisation task in a much more efficient way.
Internationalisation has the potential to be extended to become a
widely accepted general practice to make SL content amenable for
TMC to the extent that most content used in electronic communica-
tions platforms are in effect internationalised. Given their potential
global reach, such content will have a greater chance of being sub-
jected to TMC without the Sender necessarily intending such an out-
come. This means that the Sender with the internationalised Message
only prepared in one language now has a better chance of getting the
meaning across different languages and cultures in the event of some
kind of language support being subsequently applied. This will fur-
ther affect the role of the Translator and also the Sender in SL as
their roles may increasingly be synchronised. In the context of locali-
sation, a Translator may be working directly with the software
developer as in a future scenario suggested by Esselink (1999) and
in some cases the translation process may change into something close
to synchronous sight translation turning out the TL content almost
simultaneously with SL content production. Extended internationali-
sation may see the Translator in some cases becoming a designer of
SL content in view of the potential Receiver in TL. For example,
with DVD localisation, internationalisation may be applied to the
audiovisual content to allow for the strategic application of subti-
tling and dubbing in combination. This will completely change the
assumption of conventional screen translation with which the source
content is not able to be modified. Such an approach may require the
Sender and the Translator to work in collaboration.

In his future scenario for the localisation industry, Esselink (1999)
proposes a database-driven dynamic model of localisation supported
by a multilingual database as well as Translation Memory. In this
model, the key is to reduce the time lapse between the content creation
and its localisation by maximising the leverage of prior translations and
by way of efficient workflow systems. Unlike MT-based real-time
services, today's localisation is not a synchronous process and this
remains as a challenge. Localisation seeks language support solutions
by optimising the use of technology and, as a result, created produc-
tive tools specifically designed for localisation tasks. The increased
applicability of localisation methodologies to a wider area of transla-
tion is seen in a widespread interest in Translation Memory systems.
The concept of re-use of prior translations in a systematic way has an
apparent appeal and a certain degree of applicability to a wide range
of commercial translation. It is likely to impact the whole cycle of
translation as Translation Memory evolves into a more generic tool
and thus an integral part of the translation process as word processing
is today. In this scenario, every translator needs to recognise the possi-
bility of his or her translation reappearing at some future time some-
where. At the same time, this could mean that the translator does not
have to translate the same sentence ever again. The concept of interna-
tionalisation and the approach based on efficient re-use of existing
translations may significantly affect the translation process in future.

5. Conclusion

This paper endeavoured to demonstrate how localisation has
become established in the modern translation business and is provid-
ing insight into emerging practices of language support. Web locali-
sation and localisation of audiovisual content on DVDs were dis-
cussed as examples of existing and potential teletranslation practices
using the TMC framework. The paper highlighted the internationali-
sation process as a significant dimension of new language support,
suggesting its potentially far-reaching consequences. It seems justi-
fied to assume that teletranslation is developing, building on today's
localisation rather than conventional translation. The author thus
argues that localisation provides a theoretical basis to the future lan-
guage support which is emerging from the nexus of language and
technology particularly shaped by the society's shift to the infrastruc-
ture based on digital communications technology. This is creating
new contexts of TMC. To this end, localisation research holds a sig-
nificant key to understanding the future direction of translation.

Endnote

1. Some DVD titles will have subtitles in the Source Language as well as in the Target Language(s).

References:

http://www.lisa.org/archive_domain/newsletters/2002/1.5/ (consulted

Newsletter 12, 1.5.

Esselink, B. (1999), “The End of Translation as We Know it”. In: Language
International 11(5).


Kano, N. (1995), Developing International Software for Windows 95 and
Windows NT (Microsoft Programming Series). Redmond, Wash:
Microsoft Press.

Karamitroulou, F. (1999), Audiovisual Translation at the Dawn of the Digital
Age: Prospects and Potentials. In: Translation Journal 3 (3). Online at
www.accurapid.com/journal09av.htm (consulted 22.03.2003).

Lookwood, R. (1998), “Language technologies and technical communica-
tion opportunities in FPS.” In: Computing & Control Engineering
Journal 9 (6), 253-56.

11 (4).

Leiden: E.J. Brill.

Multilingual Matters.

Communication: Facing the challenges of globalisation and localisation.
Clevedon: Multilingual Matters.

Shannon, C.E. & W. Weaver (1949), The Mathematical Theory of

Minako O’Hagan is a lecturer at Dublin City University, Ireland,
where she teaches translation technology and Japanese. Her current
research project investigates the applicability of CAT (Computer-
aided Translation) tools to translating subtitles for DVD audiovisual
content. She is the author of The Coming Industry of Tele-trans-
lation (1996), and co-author, with David Ashworth, of Translation-
The Localisation Outsourcing Decision: How to

John Papaioannou, Bentley Systems, Inc.

The level of localisation outsourcing in the software industry typically varies between outsourcing all translation, engineering, and DTP, to outsourcing only translation and DTP, and keeping engineering in-house. In both cases the project management function is duplicated on both the customer and vendor side.

The first solution is most commonly found in applications with simple engineering, and well documented, independent processes. It requires fewer resources for handling the projects but more resources for structuring them. The second allows for much more complexity on the customer side, and project handling need not be as structured.

Each alternative may be appropriate to an organisation, given its structure, culture, priorities and context. A decision model can help with the decision, helping evaluate the potential advantages and of each alternative, and can provide insight in the tradeoffs involved, and reveal weaknesses and risks of the chosen alternative.

1. Introduction

The potential advantages of outsourcing are:

- Efficiency, improvement of operational performance, including cost (McFarlan and Nolan, 1995; Embleton and Wright, 1998; Hiemstra and van Tilburg, 1993; Akomode et al., 1998; Downey, 1995; Lonsdale and Cox, 2000; Blumberg, 1998; Lankford and Parsa, 1999), speed (Brown, 1997; Lonsdale and Cox, 2000; McFarlan and Nolan, 1995; Embleton and Wright, 1998), quality (McFarlan and Nolan, 1995; Embleton and Wright, 1998; Akomode et al., 1998), dependability (Embleton and Wright, 1998), and flexibility (Embleton and Wright, 1998; McFarlan and Nolan, 1995; Brown, 1997; Hiemstra and van Tilburg, 1993; Fill and Visser, 2000; Downey, 1995)

- Strategy, flexibility to redefine the organisation (Peters and Waterman, 1982; Winkelman et al., 1993; Downey, 1995; Quinn et al., 1990; Akomode et al., 1998; Embleton and Wright, 1998; Lonsdale and Cox, 2000)

- Image, how the operation looks in the books (Lonsdale and Cox, 2000; Beulen et al., 1994; Downey, 1995) or to the stakeholders (Embleton and Wright, 1998; PA Consulting Group, 1996)

- Human resources and politics, to enhance someone’s career or to reduce conflict (McFarlan and Nolan, 1995; Beulen et al., 1994; Embleton and Wright, 1998).

The main disadvantages and risks are:

- Cost escalation, due to management overhead and vendor profit margin (Embleton and Wright, 1998; Akomode et al., 1998; Downey, 1995; Lonsdale and Cox, 2000; Lonsdale, 1999). Deterioration of the quality of service resulting from the vendor assigning the best resources to other business (Terdiman, 1996; Downey, 1995; Embleton and Wright, 1998; Lonsdale and Cox, 2000).

- The reduction of the strategic flexibility from hollowing, losing over time the skills necessary for the outsourced tasks, and subsequent dependency on the vendors (Downey, 1995; Lonsdale and Cox, 2000; Embleton and Wright, 1998; Downey, 1995; Lonsdale, 1999)

- The potential impact on human resources, where outsourcing creates redundancies or limits the careers on the customer side (Papaioannou, 2002)

- The risk of loss of the opportunity to re-engineer. Once any activity is outsourced, the priority in-house tends to focus on retained activities (Papaioannou, 2002)

2. The Outsourcing Decision

The level of outsourcing most appropriate to the organisation depends on a variety of factors, which even when common between organisations, will often carry different weights and therefore lead to a different recommendation.

To facilitate the decision, a decision model such as the Simple Multi-attribute Rating Technique (SMART) (Edwards, 1977) or Analytic Hierarchy Process (AHP) model (Saaty, 1990), can be used. A software product such as Criterium® DecisionPlus® by InfoHarvest Inc. provides both options.

The criteria need to be established based on the goals of the decision, the organisational context, and the stakeholder issues. Disadvantages and risks need to be represented in the model, to ensure the decision takes a balanced view. So the criteria will include both goals and constraints. The criteria selected may be structured in multiple levels, although not necessarily a tree structure for the AHP model.

Weights to the criteria will likely require multi-point feedback, representing the stakeholder interests, including the organisation, the customers of the process, and the localisation team. These weights can be assigned on user-specified scales, and each alternative then needs to be evaluated for each criterion.

Uncertainties can be factored into the decision, by specifying an uncertainty distribution for any score. The decision scores in conjunction with the sensitivity analysis will not only point to a decision, but aid in better understanding the risks and tradeoffs.

For our organisation, the primary goal was to increase throughput, while maintaining quality and cost constraints. The stakeholders for the performance objectives were the internal customers of the localisation operation, the stakeholders for the HR criteria were the localisation team itself, and the stakeholders for the strategic criteria were the organisation itself.

The stakeholders were asked to evaluate the relative importance of the criteria that affected them. For the regional offices,
the internal customers of the operation, the scores themselves were then weighed based on their budget.

The results based on the assigned weights and scores are (Figure 1):

- **Outsource:** 0.626, the recommended alternative given the highest score
- **Expand group:** 0.500
- **No change:** 0.483

Including the “no change” alternative helps clarify the failings of the current state, as well as the disadvantages and risks introduced with the alternative chosen.

The criteria scores of each alternative provide an overview of the advantages and disadvantages of each alternative. In this case, the recommended alternative is full outsourcing, and while it is expected to improve throughput, dependability, flexibility and the HR scores, it points to reduction of the strategy score, which then needs to be managed (Figure 2).

In summary, the vendors are left to manage the projects day-to-day, and the customer staff is responsible for mid- and long-term goals, in addition to acting as enablers: acting from the sidelines, they provide the vendors with the information they need to do their job.

Vendors access directly the source code control system, retrieve the previous translation, update it, seek linguistic approval with the regional reviewers, and put the updated translation back in the source code control system.

Because the goal is to make the vendors as independent as possible, structuring the communication process and providing up-to-date information was critical. A password-protected Extranet was set up, providing information such as the contacts matrix including phone number and time zone of the contacts, the source materials schedule, product dependencies for installation and leveraging, generic and product-specific instructions, temporary license files, and various utilities.

Vendors post technical queries through a newsgroup, and cc the appropriate contacts so that the query is always “pushed” to the primary recipient. Queries are addressed on the newsgroup. Any information with long-term value is collated into the...
appropriate document in the Extranet. Further vendors cc each other in their technical queries, working essentially in collaboration.

4. Actual Benefits

4.1 Customer

With the elimination of the project management bottleneck, initial results show doubling of capacity. With a staff of two, the throughput is expected to reach 230 projects in 2003.

Free from project management, process performance can now receive more attention. For example while in the past schedule performance was only measured vs. the target release date, twelve milestone dates are now tracked for all projects, to ensure that attention is focused on the more important bottlenecks. Further customer performance is now measured systematically, to ensure that the increased capacity offered by vendors is not still hindered by internal bottlenecks.

4.2 Vendors (companies)

Vendors have increased visibility and prestige within the customer organisation. Their value-added is significantly increased from deeper integration and customer understanding. This offers opportunity for additional services, for example consulting. Further it becomes very difficult for any new vendors to compete.

Because of the increased discretion in scheduling, vendors have improved opportunities for capacity planning. The NLM provides the vendors with an important opportunity for organisational learning, and a solid reference account when they pursue similar arrangements with other customers.

The vendors are working in a collaborative relationship, cross-checking leverage statistics and helping each other overcome technical difficulties. Originally counterintuitive, the collaboration was virtually automatic when it was clarified that the customer had a “two-vendor policy” and that no matter the performance, no vendor would ever be assigned all languages.

4.3 Vendors (people)

The vendor teams have found the new set-up very motivating. The expanded scope of responsibility challenges them to grow professionally, and allows them to use a wider range of skills. Increased autonomy is also associated with the new roles. Instead of being asked “here is the source; when can you deliver it translated by?” the question is “we need to release this product up to x days after the English release; you can start when you like”, reschedule and rearrange priorities of concurrent projects at will, as long as the product is released by the target delta”. Of course there are also cost targets to be met so that early start does not increase rework unnecessarily.

5. Tradeoffs and risks

5.1 Hollowing, dependency and reversibility

The reduction of the strategic flexibility through hollowing and dependency is a key risk, which however had already become a problem as the number of products and technologies proliferated. To reduce the risk from dependency, two main vendors are used in parallel, each handling a group of languages. Because their work does not have interdependencies, they have complete control of their work, and the responsibilities are clear.

Whenever the two vendors handle languages with the same or similar legacy translations, the leverage statistics and proposed work volumes (but not the work unit rates) are compared. This serves to avoid two problems. First, even when their internal processes are different, matching leverage statistics shows that the processing variables are similar. In cases of differences the reason was either incorrect processing of the source files (for example, trying to leverage unresolved instead of resolved SGML files) or different settings during alignment. Second, inconsistency in engineering or testing hours revealed dramatic differences in work scope. This has been an opportunity to specify exactly the testing steps needed, to ensure that all required testing takes place and no more.

One factor inherent in the reduction of risk is the use of vendors with whom there is a long-standing relationship. This is 7 years with one vendor and 10 years with the other. Shared processes are being developed for the vendors, to ensure that the model is protected from personnel changes, and to provide uniform service.

Further, while the reversibility of this model is reduced, one vendor can act as a backup if the other is having trouble.

5.2 Operational performance risks

If outsourcing allows for increased capacity without proportionate increase in internal resources, the cost can be seen as reduced, at least proportionally to the increased capacity. However it should be recognised that outsourcing results in some cost increase. First, if the vendor work increases, so will the cost. The assumption is that the cost will increase less than the work, or that the freed up resources will be put to better use, such as productivity improvements. Second, outsourcing requires management. Relationship management, face-to-face meetings, and increased coordination imply additional costs, including travel and relationship building.

Refocusing at least some of the freed resources on vendor management and productivity improvements can help offset the increased cost.

To protect the organisation from deterioration of service, two vendors are used in parallel, handling similar projects. Vendor performance can then be compared, and realistic targets for improvement may be set.

Process documentation needs to happen before the tasks are outsourced, and the documentation needs to be kept up to date. This can help both standardise the service over time and across vendors, and can help reduce the learning curve of new vendors.

5.3 Human resource risks

Whether redundancies are created or not, human resource planning is necessary. For remaining employees training may be needed as preparation for their changed roles, and for any redundancies both retraining and redeployment may be needed.

Because of the multiple relationships, there is an increased possibility of difficulties due to personalities. To ensure that problems are avoided before they happen, vendors include in their weekly status report, an evaluation of the relationship of each member of their staff, in contact with the customer. The rating is aggregated in a database, watching out for trends in emerging difficulties specific to one person upstream or downstream.

Because of the emphasis on relationships, any time a resource changes relationships (including credibility and trust) need to be rebuilt. This is a bigger challenge than just replacing the skill set and knowledge of products and technologies. To facilitate, the customer care manager and the vendor manager mediate for this, primarily through the organisation of face-to-face meetings.

5.6 Loss of opportunity to re-engineer

Re-engineering and productivity improvements are instrumental in the new set of client responsibilities. In fact, with the
New Localisation Model, attention can now be focused on re-engineering systematically. These can help with cost control, maintenance of quality of service and increase of strategic flexibility.

6. Conditions for success

Outsourcing is not suitable to all clients and vendors. Key factors for success are:
- Suitable company culture
- Senior management support
- Availability of suitable vendors
- Long-term commitment
- Relationship management

The organisation must be flexible enough to change, because outsourcing requires a culture shift, including the willingness to work with interdependencies and to take risks.

While some costs, such as fixed internal resources, will be reduced or avoided, some other costs such as communications and travel will increase. The organisation needs to be able to see these in the context of direct cost savings, and the costs which would have been necessary should outsourcing not have been chosen. Furthermore, the benefits of outsourcing can be non-financial, such as improvement of the quality of service.

Vendors need to have a track record with the client, because increased outsourcing is more demanding on the skill sets on the vendor side, and the commitment required on the vendor side to make this work. Investment in relationship building is key, and vendors should be willing to refocus on a relationship level rather than a project level.

Long-term commitment is required from both parties, as responsibilities and processes change. The longer and deeper the outsourcing, the harder it is to reverse the arrangement.

Management does not stop when outsourcing starts. Outsourcing needs constant management, so that it stays on track. The focus shifts from the project level to longer-term performance metrics, but management does become even more important.

7. Conclusion

Outsourcing can increase operational efficiencies, whose benefits outweigh the risks to strategic flexibility and human resource development.

Outsourcing is not optimal for everyone. Each organisation needs to develop its own set of criteria, rate them according to its strategy and stakeholders interests, and manage both the relationships and the risks.

8. References

Akomode O.J., Lees, B., and Irgens C. (1998), Constructing customised models and providing information to support IT outsourcing decisions, Logistics Information Management, Vol. 11, No. 2, MCB University Press


Fill, C. and Visser E. (2000), The outsourcing dilemma: a composite approach to the make or buy decision, Management Decision, 38/1, MCB University Press

Hiemstra, G. and van Tilburg, J.J. (1993), Inzicht in uitbesteding: ondernemingsstrategie en besturing, Van Gorcum, Assen

InfoHarvest (2002), Corporate web site http://www.infoharvest.com


John Papaioannou has an MBA from the University of Warwick, and over ten years experience in the localisation industry. He is responsible for the localisation and internationalisation strategy, business performance management, vendor management, and process and technology management in Bentley Systems Inc., in Paris. John can be reached at john.papaioannou@bentley.com
Adopting Standards-based XML File Formats in Open Source Localisation

By Asgeir Frimannsson and James M. Hogan,
School of Software Engineering and Data Communications,
Queensland University of Technology,
Brisbane, Australia.

In recent years, major localisation vendors and key standards organisations have agreed on open XML-based standards for storage and exchange of data in the localisation process: The Translation Memory eXchange (TMX) file format for exchanging translation memory data, the TermBase eXchange (TBX) format for terminology exchange, and the XML Localisation Interchange File Format (XLIFF) for extracting and storing locale-dependent resources in a common file format. Up until recently, very few open source tools have supported these formats, and hence very few open source projects have adopted them in the localisation process. A majority of open source applications continue to use Gettext and the Portable Object (PO) file format. This paper evaluates the case for adopting XLIFF in localisation processes currently based on the Gettext toolkit, and examines the usefulness of other standards such as TMX, TBX and Translation Web Services (TWS) in these processes.

1 Introduction

Localisation of software in open source projects is usually handled by Gettext, a set of tools from the GNU project. This toolkit contains tools for extracting and merging messages from source code for localisation, as well as libraries for loading the translated messages from resource files at runtime. Gettext uses its own file format, the Portable Object (PO) format, for storing resources in the localisation process. With open source desktop environments — such as GNOME and KDE — today having translation teams for over 80 different languages, it is evident that Linux and open source software (OSS) is reaching out to a global market, and is not limited to English-speaking cultures and communities. As industry and governments around the world continue to embrace open source, localisation has become a critical factor in OSS adoption. Thus, it is increasingly important that OSS localisation processes become aligned with industry standards — allowing seamless integration of commercial and open source processes when developing for open source platforms.

In the main contribution of this paper, we present the results of working with open source contributors in defining an XLIFF Representation Guide for Gettext PO — a work now submitted to the XLIFF Technical Committee in line with their programme to develop canonical XLIFF profiles for common file formats. We propose a bridge between current localisation practices in open source and established localisation industry standards. As open source localisation is based around the common PO resource format, this paper will focus on XLIFF as a possible replacement for PO as the common format in the OSS localisation process; further, we provide standards alignment through a canonical representation of the PO file format within the XLIFF standard. XLIFF was designed principally to address needs in commercial localisation processes, especially the localisation of Microsoft Windows-based resource formats. This research aims to identify aspects of Gettext and other PO-based localisation processes for which XLIFF lacks support, and to address these deficiencies.

Building on this foundation, we then examine whether the adoption of other localisation standards may further improve OSS localisation practices. This evaluation considers the handling of terminology, translation memories and localisation workflows — areas that have that have up until now only been addressed on an ad hoc basis in open source localisation. We will investigate how successfully standards such as TMX and TBX might be incorporated into OSS localisation processes, and discuss the need for a service-based architecture, evaluating emerging standards such as TWS in open source localisation.

We have chosen to provide detailed coverage of the case for XLIFF adoption, but only a high-level overview of the case for other standards such as TMX, TBX and TWS. This is a natural focus at this time, as XLIFF is currently the only format for which there is an existing ‘equivalent’ in OSS localisation processes. Usage of Translation Memory technologies, structured Terminology Databases and service-based architectures is very limited in the open source community, as open source tools for these technologies do not exist at present. Thus, when we present models incorporating these standards, we therefore build on best practices from the wider industry and not current practices from OSS localisation alone.

This research is a first look at how open source can benefit from standards-based localisation formats; hence, much of the focus of the paper is to identify areas needing further research. Nevertheless, this paper builds a solid foundation for further research, providing a solution for representing the PO format in XLIFF, and arguing the merits of XLIFF as a common resource format in all open source localisation processes.

This paper is organised as follows: in Section 2 we provide a thorough survey of the relevant localisation and internationalisation standards, and current practices in open source localisation. Section 3 presents our main result, a canonical mapping of the PO format to XLIFF. We then look at how this format can be incorporated into existing build systems and development processes (Section 4). Section 5 builds on these foundations, discussing the need for other exchange standards in open source localisation. We conclude, in Section 6, by summarising the findings of this paper, and by examining the future of OSS localisation.
2 Background

2.1 Open Source Localisation

The term ‘open source’ relates to the practice of freely sharing access to the source code of a product, allowing anyone to extend or modify a piece of software. The open source development process has been described as a user-driven, just-in-time approach, driven by a global developer community (Berglund and Priestly, 2001). Development is driven by demand for the product within this community, and new features are implemented as a result of requests from the user base. Raymond (2000) describes the process as a ‘bazaar’ where software is released early and often, the software process is open and transparent and work is delegated as much as possible. In contrast, traditional software is developed by closed teams having long release cycles and is only released after being thoroughly tested. Surveys reveal that fewer than one in five open source participants are paid for their involvement (Hars and Ou, 2001). These characteristics imply that localisation of open source software is an ongoing process, driven primarily by voluntary efforts, in response to constantly changing source code.

Localisation of software messages is handled in most open source projects by Gettext — as specified in the Open18N Globalisation Specification (Free Standards Group, 2003). The Gettext library is based around two file formats:

- The Portable Object (PO) file format: a simple string table for storing translation units in the localisation process
- The Machine Object (MO) file format: a binary representation of a string table — used by an application to retrieve translated strings at runtime

Whilst the majority of open source projects use Gettext for internationalisation support, several projects do not rely on this toolkit. The Mozilla project (which includes the popular Firefox web browser) and the OpenOffice.org office suite use custom resource formats, and many projects rely only on the platform support provided by Java and .NET.

2.1.1 Gettext and the PO format

According to GNU Standards, software, including software messages, is to be written in American English (Free Software Foundation, 2004). By default, the original software messages are not externalised to resource files, but stored in source code, thereby allowing the application to run in the default language (English) without needing any resource files. The Gettext toolkit is then used to extract strings marked for localisation from source code, as depicted in the following diagram:

Gettext provides mechanisms for updating and merging string tables, much needed in open source development processes characterised by short release cycles and rapid change.

![Figure 1: Typical Gettext-based localisation process](image)

Gettext tools extract localisable strings from source code into PO string tables (1). Normally, each application only uses a single String table, easing the work for translators in the localisation process. Translators then translate (2) these files using PO-based localisation tools. After translation, the PO files are converted to binary Machine Object (MO) resource files (3). Applications then retrieve (4) translated strings from the language-specific binary MO files at runtime. When no translation is available for a given string, the original (English) string is used.

As depicted in Figure 2, Gettext tools first extract localisable strings from source code into a PO string table (1). Next, the newly extracted PO string table is merged with an existing PO string table containing translations (2). In addition, new entries are matched against entries of a PO Compendium — a string table acting as a translation memory, holding translations combined from multiple PO Files. Translators then translate and review changes in the updated PO string table (3) before the PO file is converted to MO for use at runtime.

While the PO format was originally intended only for software messages, many open source projects have taken advantage of the tool support for the format in the localisation of other content types, including DocBook-based documentation. There is no standard for the translation of such files; different projects use different approaches for various content types in an ad hoc fashion. In KDE and the Fedora Linux distribution, translatable segments are extracted to PO files by custom filters, as shown in the following diagram:

![Figure 2: Gettext-based workflow — merging translations](image)

Here, DocBook files are converted to PO for translation (1). After translation (2), files are converted back into language-specific DocBook documents (3). By using this approach, translators can use the same tools for translating software messages as they do for translating documentation. Custom filters are also used for other file formats, including XML-based User Interface Dialogs and Desktop Entry files, where translatable segments are added to the applications PO file in the localisation process, and merged back after localisation.

The PO file format lies at the heart of Gettext, and has three different variants: PO Template files (POTs), Regular bilingual PO files, and PO Compendium files. Entries are extracted from source code, using the Gettext ‘xgettext’ tool, and stored in POT files. POT files are initialised (using the ‘msginit’ tool) or merged with existing translated PO files (using the ‘msgmerge’ tool) and stored in a language-specific PO file, which are then in turn used in the translation process. In addition, translation entries from across projects are stored in PO Compendiums, providing simple translation memory abilities.

Each PO file generally maps to a Gettext domain, where a domain is a collection of unique translation units stored in a single MO file. Most commonly, and a de facto standard in projects such as GNOME and KDE, each application uses a single Gettext domain,
the name of that domain being generally the same as that of the
application.
The following example shows the basic structure of Gettext PO
files:

```plaintext
msgid "You have %d file"
msgid_plural "You have %d files"
msgstr[0] "Du har %d fil"
msgstr[1] "Du har %d filer"
```

Figure 4: PO file format structure

Each PO file contains a set of translation units, each having a source (msgid) and target (msgstr) field, where the first translation unit generally contains header metadata. Each translation unit can, in addition to an original and translated string, include translator comments, extracted comments from source code, source code references (source filename and line-number) and Gettext-specific flags.

PO translation units support parameter-determined plural form translation units. These translation units contain the singular English form in the msgid field, and the plural form in the msgid_plural field. As the target, these translation units have an array of type msgstr, representing the number of forms in the target language:

```
Plural-Forms: nplurals=3; plural=n==1 ? 0 : n%10<=1 : n%10>1 && (n%100<10 || n%100>20) ? 1 : 2;
```

Figure 5: PO Plural Forms

The target language may have one or more forms (Japanese has one form, while Polish has three), and the selection logic is defined in a PO header field, where nplurals defines the number of forms and plural defines a C-language expression for evaluating which item in the msgstr array to use at run time:

```
Plural-Forms: nplurals=3; plural=n==1 ? 0 : n%10<=1 : n%10>1 && (n%100<10 || n%100>20) ? 1 : 2;
```

Figure 7: Polish Plural Forms header

The selections are made using standard C ternary conditionals of the form: “condition ? true_value : false_value”, where condition is a Boolean expression. In the second example, condition is “n%10<=1 : n%10>1 && (n%100<10 || n%100>20)”, which if true yields a result index of 1, and if false an index of 2. At run time, Gettext will use the msgstr associated with the index returned from this expression.

Another unique ‘feature’ of Gettext and the PO format, is that it uses the source string (msgid) as the primary identifier. This is different from other common resource formats such as Java Resource Bundles (Figure 8) and .NET Resources, which use some sort of logical key to map to the actual source string.

Although use of the source string as id makes it easier for developers, as they do not have to handle external resource files, there are some disadvantages with this approach. First of all, there cannot be multiple translations of the same string within the same Gettext domain. This has led to ‘hacks’, where developers prefix the string with some context information, and this context information is then removed at runtime. In addition, if the original (English) string changes slightly because of spelling mistakes, fuzzy matching is required if tools are to update the translated PO files.

Source string-based ids are a challenge when updating the resource files after the source code has changed. If a string message has changed in the source code, there is no way for the tool to know which translation units are affected, and so existing translation units are simply discarded and new ones created. This makes change-tracking and version control on a translation unit cumbersome and error-prone, relying upon fuzzy string matching when a key would prevent no such concerns.

If source strings are used as ids, there is also no way to support different translations of similar strings in different contexts within the same catalogue. Some projects (for example KDE and the GNOME GLib library) combine the context and the source string to form unique ids, and have custom functions to extract the context information from the source string at runtime. Recent development versions of Gettext have however added an extra message context (msgctx) field to the PO format — supporting different contexts for identical strings.

2.1.2 Tool support in open source localisation

Open source localisation tools are mainly based around the PO file format. The most advanced of these is KBabel from the KDE project (Diehl et al., 2002). KBabel supports advanced features such as handling of plural forms in PO files, internal translation memory, dictionaries and catalogue management (Costin and Kiefer, 2004). Other popular PO editors include GTranslator, Emacs PO Mode and poEdit.

Each translation project (for example KDE, GNOME and Fedora) has its own set of web-based status pages, displaying statistics and information relating to the translation process. Recently, there has been a shift towards web-based translation portals, where contributors can, in addition to viewing statistics, also translate using browser-based translation workbench clients. The more sophisticated of these are Pootle and Rosetta (Figure 9). While Pootle is an independent project, covering translations from projects like OpenOffice, Mozilla and KDE, Rosetta is the translation portal for the Ubuntu Linux distribution.

Figure 8: Java .properties fragment — use of logical ids

Figure 9: Web-based translation portals such as Ubuntu’s Rosetta significantly reduces the technical skill needed to contribute.
The Gettext tools have some support for translation memory through PO compendium files (Free Software Foundation, 2002). These files are of similar format to PO files and contain previously translated entries. PO compendia are used to ensure consistency across applications and projects by leveraging previously translated segments. KBabel provides an internal translation memory as well as support for PO compendia and TMX — although TMX support is limited to retrieval.

Present Gettext-based localisation processes suffer from very limited translation reuse. This is mainly due to limitations of the PO format, providing little support for segmentation and abstraction of inline codes.

There are no agreed standards for handling terminology in open source localisation, and different language teams and projects use different approaches. Most commonly, teams share a glossary list, which sometimes contain definitions of terms, but usually only contain the original English term and the translated term. These glossaries are shared through collaborative wikis and web pages, and tool support is limited or non-existent.

As previously mentioned, most active open source projects have short release cycles. Localisable data is stored together with source code in version control systems such as CVS (Concurrent Versions System) and SVN (Subversion). Batch jobs take care of updating PO files from the original sources as updated source code is stored in the versioning system (KDE.org Team, 2004). Translators can view the status of translations through user friendly status pages on the web which are automatically generated from the PO files stored in the repository (Diehl et al.).

2.1.3 Limitations

Localisation of open source software is more or less limited to translation of software messages. Other aspects of the user interface such as icons, images, button sizes and layout are currently hard to localise efficiently (Van Schouwen, 2004). This is largely due to the development model adopted by most open source projects, in which the application is distributed in US English, and additional language packs can be downloaded, each containing the Gettext message catalogue in the desired language.

Though widely used and accepted in the open source communities — and successfully deployed in activities well outside its original brief — the simple string table PO file format is reaching its limits, being unable to support the rich structures inherent in modern localisation. More specifically:

- PO lacks support for advanced metadata, pre-translation and workflows
- PO-based localisation is limited to translation of textual content such as documentation and software messages, and the format does not support icons or binary images
- While PO has been successfully used for localisation of DocBook documents, the format is not a natural fit for the localisation of paragraph-based text. In particular, the format has no support for segment-based translation reuse, whereas support is very limited, and inline tags cannot be extracted

In addition, localisation of open source software and documentation is highly technical, and translators presently need advanced technical skills to be able to contribute to the process.

2.2 XLIFF and Localisation Industry Standards

The XLIFF Technical Committee (2003a) describes the purpose of XLIFF as "to define, through XML vocabularies, an extensible specification for the interchange of localisation information." As shown in the following diagram, XLIFF enables tool vendors to focus on a single file format in contrast to traditional localisation tools which have to support a multitude of native formats.

Vendors can create XLIFF filters for their proprietary file formats, extracting localisable data to the standard XLIFF format. In the extraction process, non-localisable material can be stored in a XLIFF skeleton file, which can then be used later by a merging tool to recreate the original translated file. With XLIFF, localisation engineers are not bound to use tools that support the vendors' proprietary format, but can use any tool that supports the XLIFF standard.

XLIFF supports custom defined workflow metadata during the localisation process. During the different stages of localisation, the XLIFF document might include data from Translation Memory, Machine Translation and Term Bases. Segments can be marked as 'signed-off', 'needing review' or similar, and documents can go through several localisation phases (e.g. rough translation, review).

Similar to TMX, XLIFF allows for abstraction of inline codes such as text markers, inline images and references (Savourel, 2003). This means that metadata and markup such as XML tags can be included in translations, allowing tools to display these as non-translatable markup elements.

XLIFF is a young standard, and since its introduction in 2002, some vendors have added support for parts of the standard to their tools, but many tools still provide no special XLIFF support — treating it as just another XML format (The XLIFF Technical Committee, 2003a). Many high-end localisation tools such as TRADOS and Alchemy Catalyst already have very good support for proprietary file formats such as Windows resources and Microsoft Word documents, so the need for XLIFF is not as great if customers adhere to these file formats.

Viswanadha and Scherer (2004) describe two distinct approaches to handling XLIFF in the development and localisation process:

1. The transient approach: Convert source formats to XLIFF in the localisation process, and only keep the source files in source control systems. On each iteration of the localisation process, an XLIFF roundtrip is needed, and XLIFF is simply a transient format used only in the localisation process.

2. The persistent approach: Keep XLIFF files in source control systems, and convert back to the original format at build time. Updated source files are merged with existing XLIFF documents, and metadata in the XLIFF files are preserved. This approach requires intelligent filters to handle updates and merging of XLIFF files.

Some further work is being undertaken to improve the XLIFF standard. The XLIFF Segmentation Sub Committee have been working on a standard for representing segmentation metadata in XLIFF to improve translation memory effectiveness (Jewtushenko, 2004), and this will be part of the upcoming XLIFF 1.2 specification. The existing XLIFF standard doesn’t specify how text is segmented into translation units and as different vendors and file formats use different algorithms for segmentation, it becomes increasingly difficult to leverage useful data from translation memory systems. In addition, The XLIFF Technical Committee is working on guidelines for canonical representations of common file formats (such as Java Properties and .NET resource bundles), promoting greater interoperability between tools.

2.2.1 XLIFF Document Structure

In this section, we provide an overview of the structure of an XLIFF document, and details of the key elements of its XML specification. XLIFF is structured as a group of <file> elements, each representing an extracted document. Each <file> element contains a <header> element for storing metadata relating to the original
document, as well as localisation metadata. Following the <header> element is a <body> element containing the localisable material, which may include textual or binary translation units. Textual data is represented as <trans-unit> elements, and binary data as <bin-unit> elements. Translation units can be grouped together using <group> elements, supporting hierarchical data structures such as menus.

```
<xliff>
  <file>
    <header>
      metadata related to original document and localisation process
    </header>
    <body>
      <trans-unit> or <bin-unit> translation-unit elements in optional <group> element hierarchies.
    </body>
    <file>
      additional <file> elements...
  </file>
</xliff>
```

Figure 11: Common XLIFF ‘shell’

Most elements and attributes in XLIFF are optional, making it possible to create very simple XLIFF files. This enables tool vendors to add XLIFF features to tools and filters incrementally.

```
<xliff version="1.1">
  <file original="myfile.ext" datatype="plaintext"
    source-language="en-US" target-language="nb-NO">
    <source id="#1">This is a <b>bold</b> <ept id="1">example</ept>.
  </source>
  <target>Ha en hyggelig dag</target>
  <group>
    <trans-unit id="#1">
      <source>How are you?</source>
      <target>Have a nice day</target>
    </trans-unit>
  </group>
  <context-group name="location data" purpose="location">
    <context context-type="linenumber">11</context>
  </context-group>
</file>
</xliff>
```

Figure 12: Minimal XLIFF document

Translation Units. Localisable material is segmented into translation units, represented as <trans-unit> elements (for textual content) and <bin-unit> elements (for binary content). As Gettext and other open source localisation solutions at present only deal with textual data, we will not explain handling of binary content in any further detail.

Each <trans-unit> element contains a <source> and a <target> element, representing the original and the translated resource. As a translator usually only handles the translation to one target language, an XLIFF document is strictly bilingual, allowing only a single <target> element for a translation unit. Other language translation suggestions (usually from translation memory or machine translation), stored in <alt-trans> elements (Figure 15), can however be included to provide guidance to translators.

XLIFF allows for abstraction of markup and other data in <source> and <target> elements. The following example shows the abstraction of the HTML <b> element within a translation unit:

```
<trans-unit id="#1">
  <source>This is a <b>bold</b> <ept id="1">example</ept>.
</trans-unit>
```

Figure 13: Abstraction of inline codes

Context Information. Additional context information for translation units can be stored in <context> elements, grouped in <context-group> elements. In hierarchical XLIFF files, it is also possible to define context groups for a set of translation units:

```
<group id="#1">
  <trans-unit id="#1">
    <source>How are you?</source>
    <target>Have a nice day</target>
  </trans-unit>
  <context-group name="location data" purpose="location">
    <context context-type="linenumber">11</context>
  </context-group>
</group>
```

Figure 14: Adding Context Information

Context groups can be used for a range of purposes, including translation memory lookups (‘purpose’ attribute set to ‘match’), translator information (‘purpose’ attribute set to ‘information’) and location information (‘purpose’ attribute set to ‘location’).

Workflow Information. XLIFF maintains processing and localisation workflow information in <phase> elements. The different phases are defined in the XLIFF header, as <phase> elements.

```
<xliff>
  <header>
    <phase-group>
      <phase name="review" process-name="Translation"
        contact-name="Joe Bloggs" contact-email="joebloggs@example.com"/>
      <phase name="pre-trans#1" process-name="TM Matching"/>
    </phase-group>
  </header>
  <body>
    <trans-unit id="#1">
      <source>Have a nice day</source>
      <target>Hvite en hyggelig dag</target>
    </trans-unit>
    <context-group name="location data" purpose="location">
      <context context-type="linenumber">12</context>
    </context-group>
  </body>
</xliff>
```

continued on next page
Figure 15: Workflow tracking through <phase> elements

Metadata. XLIFF provides a large set of standard attributes to support metadata such as restrictions on string length and translator comments, as well as specifying data type and resource type for a translation unit.

Extensibility. XLIFF can be extended using custom namespaces, and the standard provides a number of extension points. In addition, standard attributes can be extended by prefixing attribute values with ‘x-‘. While XLIFF extensibility is valuable, it is recommended that standard specification-provided elements and attribute values be used, enabling greater tool interoperability.

2.2.2 Translation Memory and Open Standards
Translation memory (TM) allows for reuse of previously translated segments.Translations are stored in a database, and as new documents are translated, similar or identical entries are suggested to translators based on these previous translations. Effective leveraging of previous translations relies upon a high degree of repetition in the projects undertaken; otherwise, only a limited degree of reuse is possible (Iverson, 2003). Successfully deployed TMs significantly reduce turnaround time on translation projects, reducing cost and time-to-market.

Translation Memory eXchange (TMX) is a mature XML standard for lossless exchange of TM data regardless of TM vendor and tool. Two levels of certification exist:

- Level 1 where no inline codes or markup are stored
- Level 2 which incorporates inline codes

A large number of vendors are now supporting the standards, and an increasing number of tools have been certified by LISA (the Localisation Industry Standards Association) (Zetzsche, 2003).

Effective leveraging of translations relies on a common segmentation standard. This has recently been accomplished through the Segmentation Rule eXchange standards (Localisation Industry Standards Association, 2004), which tightly coupled with the TMX standard provide a complete standards-based mechanism for exchanging translation memory data.

Future work to improve the TMX standard includes support for XML schemas (Localisation Industry Standards Association, 2004), allowing for extensible non-TMX markup inside TMX documents through the use of XML namespaces.

The internet has opened up possibilities for global sharing of TMs. (Levitt, 2003) predicts internet-shared TMs will become a standard, as this may allow higher overall consistency and shorter turnaround time compared to traditional TM systems in which each translator has their own local TM.

TMX and XLIFF make decoupling of translation tools and TMs possible (Foster, 2004b). XLIFF documents can be fed with TM entries before they are sent to translators by inserting alt trans elements. Translators will then not need access to a TM since all relevant information is stored in the XLIFF document. When translation is complete, the translated entries can be included in the TM by converting the XLIFF document to TMX using a standard XML-based transformation script.

2.3 Advantages of XLIFF and TMX over PO
2.3.1 Metadata
XLIFF allows for various types of metadata to be stored in documents. While PO files allow a limited amount of metadata (e.g. general comments and references for each individual segment, various header-specific metadata), XLIFF has an extensive range of possible elements to specify for each translation segment as well as for the XLIFF file as a whole (The XLIFF Technical Committee, 2003b). By using maxwidth and minwidth, size constraints can be specified for strings as well as other measurable elements such as images. Each segment and phase can have a translator associated with it, thus providing direct support for workflows and integrated version management. If PO files are used, this information is only available in CVS logs.

2.3.2 Workflows
The PO format does not allow for workflows beyond marking entries as fuzzy for later revision. In contrast, XLIFF specifies elements and attributes for workflow information, and localisation can pass through multiple, defined phases (e.g. pre-translation, rough translation, review), in which changes are documented in the XLIFF sources (The XLIFF Technical Committee, 2003a). This can be utilised in, for example, release planning and quality assurance processes in open source projects. Workflows can also integrate translation memory as explained below.
2.3.3 Translation Memory Improvements

XLIFF and TMX offer the possibility of shared translation memories in open source projects. At present this is hard to accomplish because of the limited information stored in the PO format. With XLIFF, multiple TM matches can be stored in the document, eliminating the need for client side TM technologies (Raya, 2004b). Sharing of TM is important to open source projects since contributors are spread geographically. Upon ‘checking out’ a file for translation, TM suggestions can be automatically inserted. When a translator has completed work and returns the file to the repository, a TMX document containing the approved translations can be automatically processed and new pairs imported into the TM.

2.3.4 Decoupling of localisation technologies

Open source localisation is currently based on the Gettext utilities, and most localisation tools in this domain are focused solely around the PO file format (Foster, 2004a). By building open source localisation tools that support XLIFF, software projects can use technologies other than Gettext without this affecting the existing localisation process.

2.3.5 XML-based processing

One advantage of XML over other file formats is the range of open and free tools and technologies available to process this format (Savourel, 2001). Many facilities exist to define parsing and transformation tools for specific formats. These can be used in the open source environment in a wide range of contexts; for example presenting summaries and data for the translation status pages of the various projects. By using simple XSLT and other XML transformation languages, intuitive summaries and reports can readily be generated from TMX and XLIFF files. These technologies can also be used by the translation tool to present user friendly reports at various stages of the localisation process e.g., printable HTML documentation can be straightforwardly rendered from XLIFF sources for comparison and proofreading.

2.3.6 Localisation of non-textual elements

XLIFF is not limited to localisation of textual content, but can also handle binary data. This opens up a lot of possibilities for future enhancements of the OSS localisation process. GNOME and KDE User Interface dialogs are currently stored in XML formats and can be encapsulated in XLIFF documents and localised using visual XLIFF tools, similar to the processing of Windows Resource files in tools such as Alchemy Catalyst (The XLIFF Technical Committee, 2003a). This however, requires architectural changes to the way localisation is handled at runtime.

3 XLIFF representation of PO

While earlier versions of the XLIFF standard provided considerable flexibility in the permissible attribute values — with many allowing free-form parsed character data — the XLIFF technical committee has recognised that this flexibility acts as a barrier to intelligent resource matching and translation reuse. Recent revisions to the standard have limited this practice, placing greater reliance on explicit enumeration of attribute values. In the subsequent discussion, there arise a number of situations in which the shell of our proposed mapping is readily defined and specified, but the resulting framework permits a range of implementations consistent with this specification. In each case we have tried to specify a model or reference implementation which we believe best represents the additional information contained in the PO format, while remaining consistent with the intent of the XLIFF standard.

The XLIFF standard and its structure were examined in some detail in Section 2.2.1, and similarly, a high-level overview of the PO format was presented in Section 2.1.1. While the details of the actual mapping between PO and XLIFF are now available from the OASIS XLIFF Technical Committee document repository, we will present here the core findings and challenges identified in the process.

3.1 Overview

We have mapped each PO file to an XLIFF <file> element, with the datatype attribute set to ‘po’ (Figure 17 below). The PO header is mapped to a translation unit, or is stored in the XLIFF skeleton. Each additional Gettext domain defined in the PO file is encapsulated in an XLIFF <group> element, with the restype attribute set to ‘x-gettext-domain’.

```xml
<file original="filename.po" source-language="en-US" datatype="po">
  <body>
    <trans-unit>
      ... PO header for default domain...
    </trans-unit>
    ... translation units for default domain...

    <group restype="x-gettext-domain"
           rename="domain-name">
      ... header and translation units for domain 'domain-name'...
    </group>
  </body>
</file>
```

Figure 17: XLIFF representation of PO — General Structure

3.2 The PO Header

The PO header contains both technical and project-related metadata, and can also contain user-defined variables. As seen in Figure 18, the header is structured as a normal translation unit, with the source field (msgid) left empty, and the PO header elements contained within the target field (msgstr). Additional metadata, such as copyright and licensing information, is stored in comment fields.

```plaintext
# Translation for MyPackage.
# Copyright (C) 2005 Foo Bar
# This file is distributed under the same license as MyPackage.
# Foo Bar <foo@example.com>, 2005.
#
msgid"
msgstr"
 "Project-Id-Version: MyPackage 1.0\n"
 "Report-Msgid-Bugs-To: foo@example.com\n"
 "PO-Revision-Date: 2004-11-11 04:29+0900\n"
 "PO-Last-Translator: Foo Bar <foo@example.com>\n"
 "Language-Team: My Language <LL@li.org>\n"
 "MIME-Version: 1.0\n"
 "Content-Type: text/plain; charset=utf-8\n"
 "Content-Transfer-Encoding: 8bit\n"
 "Plural-Forms: nplurals=2; plural=n>1;\n"
 "X-Custom-Variable: value\n"
```

Figure 18: PO Header

PO supports a set of project metadata for which there is no existing XLIFF equivalent. XLIFF is merely interested in the localisable content of a file, whereas PO also contains metadata about the localisation project and technical metadata providing context.

In processes where XLIFF is simply used as an optional localisation format (Section 2.2), it is important that translators are able to change some of the information stored in the PO header —
such as adding their name as a translator and changing the PO revision date. In addition, translators must be able to retrieve information such as the email address for bug reports. To accommodate this, the easiest and most convenient approach is to store the PO header data in a `<trans-unit>` element, as shown in the following example:

```xml
<trans-unit id="#1" restype="x-gettext-domain-header">
  <source>
    Project-Id-Version: MyPackage 1.0
    Report-Msgid-Bugs-To: foo@example.com
    POT-Creation-Date: 2004-11-11 04:29+0900
    PO-Revision-Date: 2005-02-01 12:00+0900
    Last-Translator: Foo Bar foo@example.com
    Language-Team: My Language LL@li.org
    MIME-Version: 1.0
    Content-Type: text/plain; charset=utf-8
    Content-Transfer-Encoding: 8bit
    X-Custom-Variable: value
  </source>
  <target>
    ... modified header ...
  </target>
  <note from="po-file">
    Translation for MyPackage.
    Copyright (C) 2005 Foo Bar
    This file is distributed under the same license as MyPackage.
    Foo Bar <foo@example.com>, 2005.
  </note>
</trans-unit>
```

**Figure 19:** PO header as a `<trans-unit>` element

While this approach provides a way to combine PO- and XLIFF-based localisation processes, it is not without limitations:

- Translators would need to know the purpose and format of PO headers even though they are using XLIFF-based localisation tools.
- The PO header is not a conceptual translation unit, and therefore treating it as one is not consistent with the XLIFF Specification.

In processes where XLIFF is used as a persistent format throughout the development and localisation process, it is possible to eliminate the need for the PO header in the localisation process. Project- and localisation-related metadata can be stored elsewhere, and the technical metadata needed for converting PO to MO at build time can be automatically generated.

### 3.3 Translation Units

Not surprisingly, we suggest mapping each PO translation unit to an XLIFF `<trans-unit>` element, with the `<source>` element representing the PO `msgid` field, and the `<target>` element representing the PO `msgstr` field. While this base-case is trivial, there are several features of the PO format that require further consideration:

**Plural forms.** XLIFF does not have any concept of pluralisation of translation units, but we suggest simulating this through utilising XLIFF's hierarchical structure of `<group>` and `<trans-unit>` elements. As shown in Figure 20, a group of plurals can be grouped in an XLIFF `<group>` element, and each plural form can then be represented as one `<trans-unit>` element within that group.

```xml
<group retype="x-gettext-plurals">
  <trans-unit id="#1">
    <source>%d file was deleted</source>
    <target>translation form 0</target>
  </trans-unit>
  <trans-unit id="#2">
    <source>%d files were deleted</source>
    <target>translation form 1</target>
  </trans-unit>
  ...
  <trans-unit id="#n">
    <source>%d files were deleted</source>
    <target>translation form n</target>
  </trans-unit>
</group>
```

**Figure 20:** Plural Translation Unit in XLIFF

The fuzzy flag. The fuzzy flag is Gettext's way of marking a translation as unfinished, and denotes that it needs review by a translator. This flag can easily be mapped to the approved attribute in XLIFF, where this attribute is set to 'no' if the fuzzy flag is present or when the PO entry is not translated.

**Comments.** PO files have two types of comments: Translator comments and `extracted` comments. The extracted comments occur both in PO Templates and PO files, and represent comments related to a translation unit, which are added by developers in the source code. Translator comments occur only in PO files, as they are added by translators in the localisation process. We suggest mapping these to XLIFF `<note>` elements, as shown in the following example:

```xml
<trans-unit>
  <source>hello world</source>
  <note from="po-translator">Translator Comment</note>
</trans-unit>
```

**Figure 21:** Handling of PO comments in XLIFF

**Source code references.** Each PO entry contains a list of references, referring to the location (source file + line number) from which the translation unit was extracted. The current XLIFF specification has some support for this concept through a set of predefined values (`sourcefile` and `linenumber`) for the `context-type` attribute in `<context>` elements:

```xml
<trans-unit>
  <source>hello world</source>
  <!-- Comments as note elements -->
  <note from="po-translator">Translator Comment</note>
</trans-unit>
```

**Figure 22:** XLIFF native support for references
Abstraction of inline codes. To ease the work for translators and to improve translation memory matches, it is beneficial to abstract non-translatable inline codes and markup inside translation units. Gettext supports a vast number of programming languages, each having their own set of rules for how variable parameters are represented. Creating filters to support all these languages is a daunting task, and we believe the best solution is to filter implementers to incrementally add support for abstraction of inline codes in the XLIFF representation of PO files. The following example shows how parameters from C source code can be abstracted:

```
<trans-unit>
  <source>hello <ph id="1" ctype="x-c-param">%s</ph>, how are you?</source>
</trans-unit>
```

Figure 23: Abstraction of inline codes when representing PO in XLIFF

Generating unique resource identifiers. Many tools in XLIFF-based localisation processes rely on a unique static identifier, the ‘resname’ attribute, for translation units. With resource types such as Java .properties files this is easy, as they use logical ids for unique identifiers. PO, however, use the English source string as the unique identifier, and this value cannot be used as the ‘resname’ attribute for translation units, because of the limitation of XML attribute values. To overcome this, we suggest using hash values to create unique identifiers, using the following two rules:

- For non-plural Translation Units, use a string hash of `domain.name + ":" + msgid. If the Translation Unit is in the default domain, use 'messages' as the domain name
- For plural Translation Units, use a string hash of `domain.name + "\" + msgid + ":plural\" + n + \"]", where n is the plural index of msgstr

Handling Escape Sequences. In source code it is often necessary to use escape sequences to represent characters such as the newline character (`\n`), the horizontal tab character (`\t`) and non-ASCII Unicode characters. We suggest representing these in XLIFF using the intended characters, and not the escape sequences. For example:

```
msgid ""
"Please Enter the following Data:\n"
"\t- First Name\n"
"\t- Last Name\n"
msgstr ""
```

Figure 24: PO entry with escape sequences

This entry should be represented in XLIFF as follows, replacing the escape sequences with the intended characters:

```
<trans-unit>
  <source>Please Enter the following Data:\n
  \t- First Name\n
  \t- Last Name\n
</source>
</trans-unit>
```

Figure 25: XLIFF translation unit with converted escape sequences

4 XLIFF and the software process

Up until now we have primarily focused on a general mapping between PO and XLIFF. While this is beneficial in itself, the success of adopting XLIFF in open source localisation depends on how well this format can be integrated into development processes and tools. In this section, we first examine the most commonly used Gettext-based localisation workflow, and then extend our discussion to consider incorporating XLIFF into open source localisation. Finally we discuss the need for open source translation tools that support XLIFF and present recent developments in the area.

Most GNU-based open source projects use automated build systems based on GNU Autotools. These systems support integration with Gettext, automatically extracting translatable strings from source-code and, in some cases, merging previously translated string-tables using fuzzy-matching. These build systems are composed of small unix tools, called from macros and build scripts. The Gettext toolkit comes with a set of tools for this purpose:

- `xgettext` — Extracts messages — from source code into PO Template (POT) files
- `msginit` — Initializes a POT file to a language specific PO file
- `msgmerge` — Merges a translated PO file with a newly extracted POT file by adding new translation units, marking obsolete entries (entries that have been removed from source code) and updating extracted comments and source code references
- `msgfmt` — Generates a binary MO file from a translated PO file

4.1 Current Gettext-based workflow

The following diagram gives an overview of how Gettext is integrated into the development process and build system:

![Current PO-based workflow](image)

1. Messages are extracted from source files to PO String table Templates (POT files) using the `xgettext` tool.
2. For each language, previously translated PO files are merged (2b) with the freshly extracted PO Template file using the `msgmerge` tool. If there is no existing PO file for the target language, the POT file is initialised (2a) for the target language using the `msginit` tool. In addition, entries are at this stage in some cases matched against PO Compendium files containing completed translations from across projects.
3. Translators retrieve PO files from the repository, and translate using PO-based localisation tools. Translated PO files are then committed back to the repository.
4. When a maintainer creates a release, Machine Object (MO) files are generated from translated PO files using the `msgfmt` tool, and included in the distributed package.
5. Applications retrieve translated messages from MO files at runtime.
While this diagram gives a good overview of the process, it is important to note that it is only a representation i.e., a common way of using Gettext in the development process, and that most projects use variations of this workflow. For example, in many projects it is common that merging of PO files (Step 2) is done by translators in the translation process. In addition, many projects use the inttool package to extract strings to PO from other file types (Desktop entries, User Interface dialogs and XML files) for localisation, and this tool also takes care of re-merging the translated entries. It is also important to note that Steps 1 to 3 are often iterated many times in a single release cycle, providing a way for translators to incrementally translate an application while it is being developed.

4.2 Optional XLIFF workflow in current processes

One approach to incorporating XLIFF in open source is to leave the present processes and workflows untouched, and simply use XLIFF as an optional file format for use by translators in the localisation process. With this approach, PO files are converted to XLIFF for translation, and then back-converted to PO when translation is complete:

A limitation with this approach is that every time the PO sources are updated (this happens quite frequently in open source development processes), new XLIFF files would have to be generated, losing the rich metadata stored in the old XLIFF sources. With this approach, PO is the persistent file format (Section 2.2), and XLIFF is simply a transient file format used in the localisation process. To overcome these limitations, we need an approach in which XLIFF is stored in the repository as a persistent format.

4.3 Native XLIFF-based workflow

In contrast to the minimalist view of section 4.2, the following approach proposes changes to the build system to allow the use of XLIFF as the persistent file format in the build systems and localisation process, totally eliminating the need for the PO format in the localisation process:

4.4 Gettext Integration

Building on the approach presented above, it would now be possible to integrate XLIFF within the Gettext toolkit, eliminating the need for custom XLIFF filters and tools in the build system and development process:

 Prototypes of po2xlf, xlf2po, xlfpoinit and xlfpomerge have been developed as part of this research, and are available through the XLIFF Tools Project (http://xliff-tools.freedesktop.org)
5. Open source Localisation and Open Standards

In this section we present possible solutions to the more challenging issues facing open source localisation — the actual localisation process. We do this by further examining common open source localisation processes (background presented in Section 2.1), and propose high level solutions based on standards such as TMX, TBX and Translation Web Services (TWS). Some of these standards are still in draft form (Translation Web Services, TBX), while others have been around for some time and have received broad industry acceptance (TMX). Common to all these standards, is the fact that they have emerged as a result of needs in the commercial localisation industry, and promote best practices in the field.

Some of the themes we discuss in this section are not revolutionary or new in commercial localisation settings (in fact, we draw examples from some of the available proprietary tools). They are however new in the setting of open source and collaborative localisation, and contribute to meeting many of the challenges facing open source localisation:

1. Limited Translation reuse.
3. High level of technical skills needed to contribute.
4. Error prone due to highly manual processes.
5. No standardised way of contributing to the localisation process — different projects use different approaches, but often with the same people contributing.
6. Limited support for quality control in the localisation process.

We will propose solutions for each of these issues. In Section 5.1 we discuss how we can increase translation reuse (1) through centralised translation memory repositories. Section 5.2 presents possible solutions for improving terminology management (2) in open source, through the adoption of standards such as TBX within an XLIFF-based localisation process. Finally, in Section 5.3 we discuss the need for service-based localisation workflows, and simplification of the localisation process through process automation and well defined communication protocols (4, 5 and 6). The main aim of this discussion is not to present the right solution (at this point we present ‘sketches’ rather than ‘full solutions’), but to identify ways in which workflows based on open standards can meet the challenges facing existing open source localisation processes.

5.1 Shared Translation Memory Repositories

As described in Section 2.1.3, the current Gettext-based localisation practices provide very limited support for translation reuse, mainly due to the limited support for segmentation and abstraction of inline codes in the PO format. However, large open source projects, like KDE, GNOME and the Linux distributions, are ideal ecosystems for centralised translation memory databases, due to the vast quantity of translatable data available and the large contributor base.

It is instructive to look at each of these large open source projects from an industry-wide viewpoint. Localisation projects are Localisation Providers; community contributors are translators ‘working’ for the provider, and software maintainers are localisation customers consuming services from the provider. Providers retrieve sources for translation from customers, and ideally, these sources should be matched against the providers’ translation memory, and sent to translators for completion. When a translation is complete, providers should incorporate the updated translation within their TM, increasing its value. This translation memory is the critical intellectual property of the provider.

However, in contrast to commercial localisation providers, open source TM assets could be shared across projects, further increasing the likelihood of a positive match. In addition, TM assets could be made available for download in the form of TMX archives, and imported into client localisation tools supporting the TMX format.

The following figure depicts a typical workflow for incorporating a centralised Translation Memory in the localisation process:

![XLIFF workflow with TMX-based Translation Memory](image)

In the above figure, XLIFF documents are retrieved from the version control repository (1), and submitted to the TM system for pre-translation (2). In the pre-translation process, exact and fuzzy matches are added to the XLIFF documents. In the translation process, the translator can then accept or reject the translation suggestions, and further translate the XLIFF documents (3). Approved entries in the translated XLIFF documents are then exported to TMX (4), and imported to the TM repository (5). Independent of this workflow, translated documents are committed to the version control repository.
Research

5.2 Terminology Management

As in the case of TM matching, little standardisation of terminology management has been undertaken in open source localisation. Warburton (2005b) identifies the advantages of early identification and definitions of terms in the localisation process. The XLIFF format provides good support for this process through the <mrk> element:

```xml
<trans-unit id="1">
  <source>Please make sure the path separator is set to '/
  </trans-unit>

<trans-unit id="1">
  <source>Please make sure the <mrk mtype="term">path separator</mrk> is set to '/
  </trans-unit>
```

**Figure 31:** Translation Unit without terms identified

Here we could identify the term “path separator” by encapsulating it in a <mrk> element:

```xml
<trans-unit id="1">
  <source>Please make sure the path separator is set to '/</source>
</trans-unit>
```

**Figure 32:** Translation Unit with terms identified

Identification of terms should ideally occur at a very early stage in the localisation process. This makes it possible for all language team members to standardise terminology prior to any translation work being done. As English is used as the source language in most open source projects, identification of terms can be done synchronously with the development effort, in a process similar to that depicted in the following diagram:

**Figure 33:** Marking terms in XLIFF documents

Here, an XLIFF document (ideally a language-neutral document) is retrieved from the version control repository (1). Terms are identified and marked in the XML document (2), before the file is again committed to the repository (3). In addition, a mechanism should exist that allows terms to be identified after a translation process is started.

The process of extracting terminology could be automated, by checking a document against a list of pre-existing terms. It is important to note however, that different projects might use different definitions for identical terms, and that such a process does not take into account new terms that are not present in the predefined list.

This could be improved by having a semi-automated process, where potential terms were automatically flagged, and a ‘review’ phase could be introduced to identify new terms and approve the terms automatically extracted in the process.

Obviously, the process of identifying terms is only valuable if definitions of these terms exist and are properly managed, which is the job of terminology management systems. Availability of such systems is becoming increasingly important in open source localisation, which involves geographically distributed contributors working on a common translation project. In open source, this is usually handled through sharing dictionaries using collaborative systems such as wikis.

Using web pages or wikis for sharing terminology databases is however not an optimal solution. The ideal situation would be to have the terminology data in a format that tools in the localisation process could easily exchange, access and use to automatically lookup terms that are identified in the source file. This provides a pressing need for the adoption in OSS of the TermBase eXchange (TBX) standard (Section 2.2.3). By adopting the TBX standard for terminology management in open source software, it will be possible to automatically include definitions for terms present in an XLIFF document, and ship this with translations. This again highlights the advantage of XLIFF over PO, and utilisation of XLIFF as a format for project bundles is already proven by commercial practice, with localisation tools such as RC-WINTRANS (Figure 34 below) using this approach.

**Figure 34:** RC-WINTRANS’ use of XLIFF project files

5.3 Service-based Localisation Workflows

In the previous two sections we described an approach for incorporating centralised Translation Memories and Terminology Management in open source localisation. Current localisation practices in projects such as the Fedora and KDE localisation projects require translators to manually retrieve PO files from version control repositories for translation. Having translators retrieve source files directly from the development repositories eliminates the possibility of automated server side processing. In this section we identify a need for a higher level of abstraction in the localisation processes — automated workflow systems — to accommodate many of the aspects we have discussed so far.

We propose a generalised model (Figure 35 below) based on interoperable communication protocols such as XML-RPC or SOAP Web Services, that allows for communication between clients (rich clients like KBabel, or web-based systems like Pootle) and upstream translation projects (such as KDE, GNOME and Fedora). This approach removes the need for translators to engage directly with technical issues such as version control systems, instead providing a more abstract service model which allows better control and management of the process.
This model is based on XLIFF files (and other project-related metadata) being stored in a version control repository — corresponding to the current practice of storing PO files in the repositories. Using this workflow, TM entries and Terminology definitions (as described in Section 5.1 and 5.2) could be added to the XLIFF archives before they are sent to translators (B and C); similarly new entries could be added to the TM before committing the translations to the repository (A).

Perhaps the most useful set of tools in current open source localisation practices are the catalogue managers (Section 2.1.2), in the form of translation status pages and tools like KBabel’s Catalog Manager. These tools present translation tasks and status reports in user friendly and intuitive ways, and in addition often provide mechanisms for direct retrieval of sources from version control repositories. Recently, through portals such as Pootle, Rosetta and IRMA, these status pages have been extended to become fully fledged localisation solutions, allowing translations to be performed in a web-based environment. Extending these concepts, we propose a web services-based interface (D) to the Localisation Infrastructure system, enabling any client supporting the protocol (such as web-based translation portals, rich client localisation tools) to communicate with the upstream localisation project.

The OASIS Translation Web Services (TWS) committee aims to develop a web services-based interface for automating communication between localisation vendors and clients. This specification includes methods for requesting localisation quotes, retrieving and submitting localisation jobs, and querying the status of localisation jobs. In addition, the specification covers queries to language service providers to determine which services and languages are supported.

Open source localisation processes based on community contribution do not fit in very well within the scope of the TWS specification. In open source projects, translators (community contributors) are the active party, providing their services by translating an application to a specific language. There is however an emerging need for standardised and automated processes in open source localisation, to allow community contributors to query, retrieve and submit translations. Currently this can be done only by manually checking the translation status pages for a project — commonly via a web browser, or alternatively (in the case of the KDE project) by using a tool such as the KBabel Catalog Manager to interact with the version control system to retrieve this information.

In contrast, a web services-based protocol such as SOAP or XML-RPC, provides considerable flexibility in implementing clients. By implementing a standards-based interface, custom ‘Catalog Manager’ applications can be used to retrieve translation projects (collection of XLIFF files), which in turn can be translated in any XLIFF-enabled editor. The same interface could then be used to develop web-based translation portals, rich client catalogue managers, and localisation suites. Currently this is a big obstacle for translation portals such as Pootle, Rosetta and IRMA, as there is no standardised mechanism for upstream translation updates.

Defining a standard interface for interacting with open source localisation repositories also opens up possibilities of composing services in a unique way. This is especially useful for community-driven Linux distributions such as Ubuntu and the Fedora Project — distributions that are composed by combining software from a multitude of projects. For example, in the Fedora project, it would be beneficial if translators could automatically retrieve translation tasks for the current GNOME or KDE release, without having to manually track each project independently.

In building an infrastructure targeting open source localisation, we cannot assert that localisation contributors have high-speed and reliable internet connections, or even have internet connections at all. In developing countries (which indeed have some of the most interesting open source localisation projects), it is often more feasible to arrange translation gatherings (nick-named ‘translate-a-thons’), where contributors come together in school computer labs or similar and translate open source software using thin client web-based translation tools, with one computer acting as a server running a localisation portal such as Pootle. The proposed model also caters for this scenario, as local servers could be set up, or sets of translation tasks could be retrieved upfront.

In addition, the proposed localisation model (Figure 35) also supports the current practice of contributors directly interacting with the version control repositories, retrieving and committing sources for translation in their XLIFF-based localisation tool of choice (E). We see it as important to support this simple approach, as it would eliminate the danger of a single point of failure, and avoid the risk of losing contributors hesitant to change their current localisation practices.

The overall aim in proposing a service-based localisation infrastructure is to enhance and simplify the localisation process from a translator point of view. The level of technical knowledge needed to contribute to current open source localisation processes acts minimally to decrease efficiency, and at worst, to turn potential contributors away. With the advent of structured approaches to translation memory and terminology management, these issues must be addressed — otherwise these technologies may introduce yet another level of complexity for the translation contributors.

6 Conclusions

In Section 3 we presented the main contribution of this paper — a bridge between XLIFF (the industry standard for exchange of localisable data) and Gettext (the de facto standard in open source localisation). This was accomplished by developing an XLIFF representation guide for the Gettext PO file format, followed by a discussion of how best to incorporate XLIFF within present open source development processes. In this section we specifically targeted the PO format, as it is being used (indeed, exploited) as a common resource format for localisation of a number of file types in open source. Our goal, however, has not been simply to eliminate PO in favour of XLIFF in Gettext-based localisation, but rather to propose XLIFF as the standard resource format for all open source localisation, in much the same way as PO is being used (or indeed, over-used) at present. This research has only fulfilled part of that goal, as only the PO format has been thoroughly covered. To make XLIFF a more attractive option in open source localisation, tools and filters to convert between XLIFF and other common OSS file formats are needed.
By adopting XLIFF as the common resource format in open source localisation, we have also managed to decouple technologies in the localisation and internationalisation process. Developers are no longer limited to using Gettext and can investigate using other alternatives without disrupting the localisation process. Such decoupling brings forth a unique opportunity for further research in the area of localisation technologies in the development process.

During the course of this research, we have noticed an increasing interest from the open source community in regard to adopting XLIFF in localisation processes presently based on Gettext and the PO format. Developers from leading localisation projects (including Ubuntu’s Rosetta, Gnome, KDE, Fedora and Poole) have expressed an interest in XLIFF as a replacement for PO. It is, however, too early to determine if, and when, these new standards will be adopted in these projects, as much of the supporting infrastructure and tools (mainly XLIFF filters and localisation tools) do not yet exist. In addition, despite the deficiencies and limitations of Gettext and PO, current localisation practices are to a large extent successful — with projects like KDE and GNOME being localised to over 80 languages, and the benefits of switching to XLIFF once will have to be overwhelming to justify such a move.

In Section 5 we looked beyond XLIFF, discussing possible future opportunities for open source localisation processes, further extending the bridge between open source and proprietary standards. The discussion was built on the foundation laid in the previous section — XLIFF as the common resource format — and focused on three main areas in need of better solutions in open source: translation reuse, terminology management, and service-based localisation workflows.

Translation reuse in open source has until now only been addressed in an ad hoc fashion, through PO Compendia and the KBabel PO editor, and there are no structures in place for sharing translation memory data. In addition, the data fed into these translation memories are not ideal, as current PO-based processes have no support for segmentation, alignment and abstraction of inline codes and markup. Adopting XLIFF and TMX in these processes provides a foundation for building quality translation memories and storing higher quality data, which can in turn increase translation efficiency in years to come. There are, however, no existing open source TMX-certified Translation Memory systems available, and further research and development is needed in this area.

Terminology management is arguably as important as translation reuse for quality localisation solutions. In current open source processes terminology is, at best, handled in an ad hoc fashion. Terms are not identified prior to the translation process, and if terminology management is undertaken at all, language teams simply use combined bilingual glossaries of words that have been previously identified in the translation process. The broader localisation industry has identified and acknowledged the importance of proper terminology management, and has agreed on a standard file format, TBX, for exchanging terminology. Further research is needed in defining workflows incorporating terminology in open source, and further, developing open source terminology management systems supporting these standards.

As with many other industries, the localisation industry is looking at ways of automating the business processes through service oriented architectures based around SOAP Web Services. This work is now being formalised through the OASIS Translation Web Services committee, developing a specification that provides interfaces for automating communication between localisation vendors and customers. As the commercial interest in open source continues to grow, Translation Web Services (TWS) can be of great benefit to open source software. Open source software vendors can use this channel to localise resources (now based on XLIFF) through commercial localisation service providers. There is, however, a need for service oriented architectures in the collaborative open source localisation environment, as this process is not covered by the TWS specification. In open source localisation, translators are the active part (the localisation provider), providing services to an open source localisation project (the client), and research is needed to define a service oriented architecture for these environments. As with industry standards in the field, the goal of this research is to provide an automated system eliminating many of the trivial and non-localisation related tasks relating to the process. In other words, the goal must be to let the translator concentrate on the translation and not the surrounding technicalities.

The standards we have covered in this paper, except for TMX, are still fairly youthful, and are still in a process of maturing. XLIFF 1.1 was approved as an OASIS Committee Specification in November 2003, but is still being finalised before submission for approval as an OASIS standard. Revision 1.1 of the standard limited the amount of free-form metadata allowed through deprecation of the <prop> element, and provided a larger set of pre-defined attribute values. As free-form data is being restricted in the XLIFF namespace, the specification now recommends using other name spaces within the document for storing additional information, further expanding the possible usage areas of the format. This has recently been demonstrated through the work on the XLIFF Representation Guide for HTML, where common HTML attributes are added to the translation units, further enriching the set of available metadata.

As the localisation industry adopts service-based architectures and Translation Web Services, we believe there will be a strong focus on the linkage between standards-based XML file formats. The beginnings of this movement can already be seen, through the inclusion of elements from the XLIFF namespace in the TWS specification, and it is reasonable to believe that there will also be a stronger linkage between XLIFF and TBX in future revisions of the standard, allowing linking of terms identified in the XLIFF document with definitions in the TBX document.

Adoption by the open source community of the standards-based file formats discussed in this paper will be of considerable benefit to OSS localisation processes, and provide an open, fertile environment within which these standards may evolve toward technical maturity. This will not only benefit the open source communities, but also the commercial localisation industry. Up until now, most implementations taking advantage of these standards have been proprietary solutions, only tested by a single vendor. Successful adoption of these standards in open source can in turn provide a playing field in which these standards can evolve and mature independently of commercial interests.

As a conclusion to this paper, the XLIFF resource format is mature and rich enough to provide a valuable replacement for the Gettext PO format. But for this to happen, development of localisation tools supporting XLIFF is needed, and, further, development of XLIFF filters for other file types that currently use PO in the localisation process. Open source localisation can also benefit from standards beyond XLIFF and, with further research, the localisation process can be enriched through standardised management of translation reuse and terminology. To fully take advantage of this, structured localisation workflows need to be developed — and these can benefit from the development in the area of Translation Web Services, providing a way of automating the localisation process.

Acknowledgements

The authors would like to acknowledge the assistance and support of Paul Gamez and the team at Red Hat in Brisbane, and all those who have contributed to this work through the XLIFF Tools project: Josep Contal, Bart Cornelis, Fredrik Corneliusson, Karl Eichwalder, Tim Foster, David Fraser, Bruno Haible, Rodolfo M. Raya and Yves Savourel.
References


Asgeir Frimanussisson recently completed an honours degree in Software Engineering at the Queensland University of Technology. His thesis on XML standards in OSS Localisation — a work on which much of this paper is based — was awarded the LRC Best Thesis Award in 2005. He will shortly continue as a PhD student in a related area, and can be reached at asgeirf@gmail.com.

James M. Hogan is a Senior Lecturer in the Faculty of Information Technology at the Queensland University of Technology, where he leads research and teaching in Software Internationalisation. His interests include XML standards and context-based matching and retrieval. He may be contacted at j.hogan@qut.edu.au.
Translation Web Services — an Implementation for the IGNITE Project

Kevin Bargary describes the Localisation Research Centre’s (LRC) involvement in the Translation Web Services specification implementation, which is part of the IGNITE project currently being undertaken by the LRC and its partners; VeriTest, Archeptypon, PASS Engineering and Vivendi Universal Games.

IGNITE will pool together linguistic infrastructure resources and provide convenient access and a marketplace for them. This will satisfy a pre-requisite for the success of the European digital content industries.

Keywords: Translation Web Services, OASIS, Localisation, Standards, SOAP, WSDL.

Introduction

IGNITE is divided into three phases, the third of which is the ‘Performance Analysis’ stage of the project. This third phase aims to firstly verify standards and their use and implementation in tools and source material, and secondly, enhance the standards themselves by providing feedback from the project findings to the various standards committees. With this in mind the LRC became actively involved in the Translation Web Services (TWS) Technical Committee (TC) which operates under the umbrella of OASIS. Our main function as members of the TC was to provide an independent implementation of the draft specification that was already in place. As this was the first draft of the specification, and it had not been previously implemented, some issues were expected — it was our job to identify these and see how the specification functioned in a practical working scenario.

Web Services

The basic premise of TWS — and indeed any web services implementation — is that you have a client machine and a server machine. The server machine will contain a pre-programmed set of methods or functions that the client machine will access using web services. An example of this would be an online credit card validation system. One website (the client) connects to a remote service (the server) with the details; the service validates the credit card and returns the result to the website.

Implementation Work

The first stage was to decide on an implementation platform. The underlying technology in web services is SOAP (Simple Object Access Protocol). SOAP can be thought of as a message-passing system between two computers using the Hypertext Transfer Protocol (HTTP) over the Internet. The TC decided to use the J2EE development platform and the Java programming language. The rationale behind this decision was that the open source ‘Apache Project’ had a Java-based implementation of SOAP called AXIS. The AXIS implementation is a reliable and stable base on which to implement Java Web Services. This implementation provides an Application Programming Interface (API) into the SOAP actions that are required for implementing the TWS specification. The logical choice of web server to complement the use of Apache AXIS was Apache Tomcat — so this was used to host the web services on our internal LRC server.

The development process was based on the prototype model of software development. The first stage was to start with one of the 18 services currently available in the TWS specification, and from there to develop one service at a time and report back to the TC on any issues or suggestions for improvements that arose as we progressed. Table 1 shows a list of all of the services available in the current TWS specification. The services are divided into three categories; namely ‘Required Services’, ‘Optional Services’ and ‘Recommended Services’.

As previously mentioned, AXIS provides an API to the SOAP functionality. It also provides two command line utilities that further aid the implementation of web services. The ‘Java2WSDL’ utility takes pre-existing Java code and creates a WSDL file for that code. A WSDL (Web Services Description Language) file indicates how the client can access the service, i.e. what parameters need to be passed to the service in order to evoke a response. The TWS specification already has a WSDL, so this utility was useless for our purposes. However, the second utility, ‘WSDL2Java’, creates the Java stubs (Java files that contain the code needed to use SOAP) required by

(a) the server to write and deploy the service and
(b) the client to access the service through its own code.

Figures 1 and 2 show this process.
Firstly the Java stubs are created:

![Diagram showing the creation of Java stubs from WSDL.]

**Figure 1: Creating the Java stubs from the WSDL.**

Then the Java stubs are used by the client application to access the services and by the server to deploy the services.

![Diagram showing connecting client and server through Java stubs.]

**Figure 2: Connecting client and server (through the Java stubs) over HTTP using SOAP.**

The first service that we implemented was the `retrieveServiceList` service. This service was chosen because there were no input parameters required for it. All that was required to invoke the service was an instance of the `RetrieveServiceListRequest` class. The `RetrieveServiceList` service returns a complete list of services offered by a particular vendor. This will include the languages dealt with and services offered by a particular vendor (Translation Web Services Specification Draft 1.0). After writing the server-side code to handle a `RetrieveServiceListRequest`, i.e. return all of the appropriate values, the next stage was to create a simple test class that could instantiate a `RetrieveServiceListRequest` and handle the results received back from the server in a `RetrieveServiceListResponse`. With both classes now ready, we needed to deploy the services to the Apache web server. The WSDL file also contains the location of the service, i.e. where it can be accessed from. Deployment is necessary to ensure the service is in the location as defined in the WSDL. When the utility ‘WSDL2Java’ creates the Java stubs needed for the server-side machine, it also creates two other files that are used to deploy and ‘un-deploy’ the service to a web server (Apache Tomcat). These files are called Web Service Deployment Descriptors (‘deploy.wsdd’ and ‘undeploy.wsdd’).

For more information on Translation Web Services visit [http://www.igniteweb.org/documents.php](http://www.igniteweb.org/documents.php) to see a presentation by Peter Reynolds entitled Web Services for Translation, given at the IGNITE Working Conference in Dublin on 14 December 2005.

The implementation is currently on [www.electonline.org:8080/index.html](http://www.electonline.org:8080/index.html)

The next stage in the development of the implementation was to write the code for the rest of the services. While writing the code we encountered some issues with the specification (including inconsistencies between the schema and the specification document). These issues were quickly amended by the TC. During the process of coding we also made some suggestions to the TC about possible improvements to the specification and we were actively involved in applying these changes. For example, the service ‘retrieveQuote’ in the original specification did not return any information about the location of the actual quote. This was deemed to be an important piece of information for this service and was promptly included in the specification.

With the code for the implementation of the services now written, the initial service deployed (`retrieveServiceList`) was undeployed and the full list of services was deployed to the web server. A JSP (Java Server Pages) client interface was developed to allow for the input of the parameters required for each service and also to show the responses from the server — see Figure 3.

**Conclusions**

Working on the implementation of the Translation Web Services specification has been very beneficial for the IGNITE project. One valuable lesson learnt during this implementation was the importance of having a reference implementation of a standard. This is vital to ensure the standard works in a practical environment. Becoming involved in the development of the TWS specification draft has allowed the IGNITE team to see how a standard comes together from initial design specification to an industry-accepted standard. The real benefit of the TWS standard will be seen when an implementation exists that incorporates the XLIFF standard for the exchange of localisable content through the localisation process. XLIFF was designed as an interchange file format for the loss-less exchange of data for localisation. If TWS is used as a medium for the transport of XLIFF, then a whole new localisation process in which automation is a key factor could evolve. Our work within the TWS TC has allowed us to be ingrained at each stage in the development of this standard. With the knowledge gained from this work we in the IGNITE project hope to research the possible marriage of these two important localisation standards in a working automated environment.

For more information on Translation Web Services visit [http://www.igniteweb.org/documents.php](http://www.igniteweb.org/documents.php) to see a presentation by Peter Reynolds entitled Web Services for Translation, given at the IGNITE Working Conference in Dublin on 14 December 2005.

The implementation is currently on [www.electonline.org:8080/index.html](http://www.electonline.org:8080/index.html)

Kevin Bargary graduated from the University of Limerick in 2004 with a B.Sc. in Computer Systems. As well as working as a researcher on the LRC-coordinated IGNITE project, he is currently pursuing an M.Sc. in the area of open source localisation — also at the University of Limerick. He can be reached at Kevin.Bargary@ul.ie
The Sinhala Collation Sequence and its Representation in UNICODE

By Weerasinghe A. R., Herath D. L., Gamage K.
Language Technology Research Lab,
University of Colombo School of Computing, Colombo, Sri Lanka.

Abstract
The alphabet of a language is perhaps the first thing we learn as users. The alphabet of our mother tongue would be the first alphabet we ever learn. And yet, a closer look reveals that there is much about such an alphabet that we have not explicitly specified anywhere. The Sinhala alphabet order is a prime example. We use it, recite it and yet would be hard pressed to define it explicitly.

Sinhala is spoken in all parts of Sri Lanka except some districts in the north, east and centre by approximately 20 million people. It is spoken by an additional 30,000 (1993) people in Canada, Maldives, Singapore, Thailand and United Arab Emirates. Sinhala is classified as an Indo-European language and used as an official language.

The UNICODE Collation Algorithm (UCA) is an attempt to make explicit the collation sequence of any language expressed in the UNICODE (or any other) coding system. In order to express the Sinhala collation sequence (alphabetical order) using UCA, the authors undertook the task of identifying unresolved issues facing the unambiguous definition of the order. This paper first describes the issues identified through this study, suggesting alternate solutions and recommending one of them. Finally, it sets out the recommended collation sequence for Sinhala in the form of the UNICODE collation specification. The outcome of this process is a unique and unambiguous expression of the Sinhala collation sequence which could be tested using existing tools and software environments.

Keywords: UNICODE, Sinhala, Collation, UNICODE Collation Algorithm, Localisation, Internationalisation.

1. Background

The Collation Order of any Language is one of the most important issues that has to be resolved urgently in the process of standardising such a language. Since languages have been used for centuries by humans without worrying about their irregularities, often their constructs are extra logical. The collation sequence of most languages faces this non-logical nature. Steps are being taken to avoid these ambiguities and irregularities and also to formalise the collation sequence as much as possible since it plays a major role in the process of standardising the languages. This is particularly so in the case of electronic texts of a language since the computer needs explicit ordering information in order to process such a language.

To illustrate simply, even in the apparently well understood case of the English Latin-1 character set, the font itself does not encode order. If it did, words beginning with ‘Z’ will precede those beginning with ‘a’ since all upper-case letters precede all lower-case letters in all encodings (including UNICODE) of English. The case for Sinhala is no exception. In fact, as will be clear in the ensuing discussion, the Sinhala collation sequence demands us to take some decisions thus far not explicitly made for the Sinhala language as a whole.

2. Introduction to Sinhala Alphabet

The Sinhala alphabet consists of characters which represent almost all the sounds that can occur in the language. On the other hand, it is phonetically over-specified in that there are multiple characters to represent the same sound: for example (dental la) & (Alveolar la), (dental na) & (Alveolar na), (voiceless ta) & (voiced ta).

The commonly accepted Mixed Sinhala Alphabet has a set of sixty characters. This set of characters can be classified into three categories, namely vowels, semi-consonants and consonants.

Vowels: There are 18 vowels in the Sinhala alphabet, w, wd, we, wE, …

Semi-consonants: there are two characters which can occur only with a vowel: x and #

Consonants: there are 20 consonants in the alphabet: l, L, …, y, <

In addition to the above characters there is another set of symbols called vowel-strokes or ‘pilli’, to represent vowel sound when vowels are combined with consonants. For example: (dental la) & (Alveolar la), (dental na) & (Alveolar na), (voiceless ta) & (voiced ta).

The relative order of these character sets is also well defined, i.e. vowels are followed by semi-consonants and consonants, except for ‘ee’ and ‘ê’. The relative order of these character sets is also well defined, i.e. vowels are followed by semi-consonants which themselves are followed by consonants.

3. Methodology

The methodology adopted in this study is to first gather existing views and traditions of Sinhala language collation from scholars, observe the collation sequence adopted by the
major standard dictionaries and to research how collation sequence is determined at various state organisations in performing their regular tasks.

It is expected that this kind of study would reveal the main issues related to the collation order of Sinhala and how these issues are addressed by scholars, in dictionaries and by organisational practice.

4. Issues identified in Collation Order

The following issues* were identified at the beginning of the study. With these issues in mind, prominent dictionaries were searched, the views and opinions of scholars were obtained, and the procedures followed by state institutions and organisations were observed. The following listing identifies five major issues (first five with associated levels of importance) and three less critical ones which need to be resolved in order to proceed with the specification of an unambiguous collation sequence for Sinhala.

**Issue # 1 [Level 2]**

The positions of ‘anusvara’ and ‘visarga’ in the Sinhala collation sequence. While this is not really ambiguous as far as many dictionaries and linguists are concerned, its place at the beginning of the UNICODE code chart makes it an issue to be resolved.

**Issue # 2 [Level 3]**

The position of the ‘hal’ sign (halant form) of a consonant in the sequence. Many alphabets of Sinhala do not explicitly specify the place of the ‘pure consonant’ form (the so-called ‘vowel removed form’) of Sinhala letters. As such, there is common confusion as to its rightful place in the alphabetical order. For digital representation, this becomes an important issue to be resolved.

**Issue # 3 [Level 3]**

The positions of words containing yansaya, rakransaya and rephaya when there are two or more alternative forms available for the same word. While in general there is agreement that these ‘short forms’ are exactly equivalent to their non-shortened forms, in a digital representation a decision has to be forced as to which of them precedes the other.

**Issue # 4 [Level 2]**

Miscellaneous issues such as the archaic way of writing words such as නන්දු and the irregular forms නන්දු and even නන්දු. Though rare, the exact function and position of such words in a sorted list of words needs to be explicitly given to facilitate digital processing.

**Issue # 5 [Level 3]**

Whether ‘cce’ is a ligature of ‘e’ and ‘ce’ or a single letter; and the right position of ‘ce’. This again is an issue raised by its existence as a separate code point in the UNICODE code chart.

**Issue # 6 [Level 0]**

The position of the letter ‘?” The Latin symbol ‘?’ was superimposed on the Sinhala letter ‘?’ before the symbol ‘?’ was introduced into the alphabet. For this reason, and the phonetic closeness of the sounds represented by ‘?’ and ‘?’, the letter ‘?’ has been popularly placed after the letter ‘?’ in many contexts. On the other hand, the letter ‘?’ being the newest letter of the Sinhala alphabet, is also placed at the end of the list of consonants in the alphabet.

**Issue # 7 [Level 1]**

There is a rule in the Sanskrit writing system that the consonant that comes after a rephaya is doubled, for example the words සාරුණයෙන, ශාරණයෙන. The reason for this appears to be to display other vowel modifiers clearly when they are used with a consonant that comes after the rephaya, for example as in එංගෙරුණාය, එංගරුණාය, එංගරුණාය, එංගරුණාය. The problem arising with this kind of phenomenon when sorting is whether their positions should be considered based on this doubled form, or on their corresponding simplest form (as එංගරුණාය, එංගරුණාය, එංගරුණාය, එංගරුණාය in the examples above).

**Issue # 8 [Level 1]**

Finally, the sort order of words which consist of intra-word spaces, for example, the name නම්බරෙන්. The issue here is whether to consider this as two words, to ignore the space and consider the string as a single word, or to consider the entire string including the space as the single full word.

While these issues were the ones pre-identified in the study, the availability of online tools for testing any suggested collation sequence expressed in accordance with the UNICODE Collation Algorithm specification, allowed us to look for any other issues which might be ‘thrown up’.

5. Observation made in Dictionaries

For the purpose of this study some of the main and popular dictionaries were selected from among the various Sinhala dictionaries published. These dictionaries were selected after considering multiple attributes such as their quality, quantity in circulation, real usage and the perceived degree of authority of their compiler(s).

The following were the dictionaries selected for the present study:

1. Sri Sumangala Shabdakoshaya compiled by Ven. Velivitiye Soratha Thero
2. Sinhala Shabdakoshaya published by the Department of Cultural Affairs
4. Prayogika Shabdakoshaya compiled by Dr. Harishchandra Wijetunge
5. Sinhala Vishvakoshaya published by the Department of Cultural Affairs

* N.B : Issues # 6, 7, 8 were not taken into consideration in the first phase of the survey. The Levels indicate the perceived severity of the issue concerned, 3 being the most critical.
The issues identified regarding the collation order of Sinhala were kept in mind while these dictionaries were being studied. The information gathered from these dictionaries regarding each issue is summarised in Table 1.

Based on the above, some partial conclusions could be made as follows:

**Issue 1:** This appears to be a non-issue as far as dictionary compilers are concerned. These two semi-consonants are placed at the end of the set of vowels in the Sinhala alphabet.

**Issue 2:** Apart from the Sri Sumangala dictionary, each of the other four had a clear decision that the ‘hal’ form comes after all other vowel derivatives. While the justification given by each of these two schools makes sense in their own contexts, the majority decision may need to be adopted for our purposes. Section 6 illustrates the difference between the two schemes.

**Issue 3:** Interestingly, none of the dictionaries are able to shed light on this issue owing to each only containing a single form – either the short or the non-short. As such this issue cannot be resolved using this methodology.

**Issue 4:** There is wide variation on the treatment of this issue. As such, a final decision on resolving this is deferred at this stage.

**Issue 5:** All dictionaries implicitly consider ‘Z’ as a ligature by their positioning of words beginning with it appearing soon after those beginning with ‘j~’.

**Issue 6:** In all dictionaries which includes it, the position of the letter * is immediately after the consonants.

**Issue 7:** The doubling of the reph-modified consonant is given as a spelling variant of the simpler form in all dictionaries which contained it.

**Issue 8:** All dictionaries include words which have intra-word spaces where appropriate.

6. Procedures Followed by Dictionaries to Sort Words

The procedure followed in the Sri Sumangala Sabdhakoshaya to arrange words manually according to the alphabetical order is best specified by the following algorithm:

![Table 1: The position taken by dictionaries on the eight issues under consideration](image)
1. Identify the syllabic units* of the two words
   Let the two words be w₁ and w₂
2. Write each syllabic unit of both words as a consonant-vowel pair†
3. i = 0
4. Choose the iᵗʰ character of each word
   Let the two characters be w₁ (ch (i)) and w₂ (ch (i))
   4.1. If w₁ (ch (i)) = = w₂ (ch (i))
       4.1.1. i = i + 1
       4.1.2. go to 4
   4.2. Else if w₁ (ch (i)) > w₂ (ch (i))
       4.2.1. w₁ > w₂
       4.2.2. break
   4.3. Else
       4.3.1. w₁ < w₂
       4.3.2. break

Dictionaries other than *Sri Sumangala Sabdhakoshaya* compare consonant-vowel pairs in a different manner which makes the two approaches different. In this method when two consonant-vowel pairs are compared two consonants and two vowels are compared separately. In the cases where vowels are not present the consonant of the next consonant-vowel pair is not taken as in the *Sri Sumangala Sabdhakoshaya*.

7. Views of Scholars/Academics and Linguists

The following scholars and academics were consulted with a view to acquiring their expert views – often based on their respective linguistic persuasions. The aim of the consultation was to attempt to achieve consensus and not just for documenting their independent views.

- Professor Vinee Vitharana (VTH)
  Chief Editor of the Sinhala Dictionary. Former Professor of Sinhala at the University of Ruhuna.

- Professor Wimal G. Balagalle (WBA)
  Former Chief Editor of the Sinhala Dictionary. Emeritus Professor of the University of Sri Jayewardenepura.

- Professor W. S. Karunathilake (WSK)
  Former Professor of Linguistics at the University of Kelaniya

- Professor J.B. Dissanyaka (JBD)
  Emeritus Professor of the University of Colombo

<table>
<thead>
<tr>
<th>Issues</th>
<th>VTH</th>
<th>WBA</th>
<th>WSK</th>
<th>JBD</th>
<th>SGA</th>
<th>HWI</th>
<th>RUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>At the end of the vowels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>At the end of the vowels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Need to have a policy; what is simple should come first</td>
<td>Need to have a policy; whatever non-confusing</td>
<td>Need to have a policy</td>
<td>Need to have a policy; what is simple should come first</td>
<td>Need to have a policy; priority must be given to what is commonly written</td>
<td>Need to have a policy; priority should be given to the tradition</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>$\varnothing$ should be written as it is</td>
<td>$\varnothing$ should be written as it is</td>
<td>Sanskrit loan words should be written in their traditional forms, but English loan words can be written in either form</td>
<td>$\varnothing$ = $\varnothing$</td>
<td>$\varnothing$ = $\varnothing$</td>
<td>$\varnothing$ = $\varnothing$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td>‘$\varepsilon$’ is a ligature $\varepsilon$</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td>$\varnothing$ comes at the end of consonants</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>- give along with the simplest form-</td>
<td>- give along with the simplest form-</td>
<td>- give along with the simplest form-</td>
<td>- give along with the simplest form-</td>
<td>- give along with the simplest form-</td>
<td>- give along with the simplest form-</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>- space has to be considered-</td>
<td>- space has to be considered-</td>
<td>- space has to be considered-</td>
<td>- space has to be considered-</td>
<td>- space has to be considered-</td>
<td>- space has to be considered-</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: The position taken by Linguists on the eight issues under consideration

* Syllabic unit means entity that contains a consonant and a vowel which is represented with pilli, e.g. $\ddot{\varepsilon}$, $\ddot{a}$, are syllabic units and in some contexts $\ddot{\varepsilon}$, $\ddot{a}$, can also be syllabic units, in $\ddot{\varepsilon}$ and $\ddot{a}$ are syllabic units. The syllabic units of $\ddot{\varepsilon}$ are $\ddot{\varepsilon}$, $\ddot{\varepsilon}$, $\ddot{\varepsilon}$, $\ddot{\varepsilon}$
† If the syllabic unit does not consist of a vowel write ‘null’ in place of vowel (e.g. $\ddot{\varepsilon}$ = ‘null’). The ‘null’ is considered as a character and it is the character that has the greatest value in the weight space.
Having explained the aims and objectives of this study, a list of lexemes that concretely represents all the possible issues was carefully designed and given to each consultant – rather than posing the issues in their abstract form. This approach forced an explicit response rather than inviting rigorous expositions of the theoretical basis for same. There were some issues which some of the linguists could not provide a direct answer to. However, most were able to make their suggestions as to how to resolve such issues by relying on their own linguistic theories. The books written by some of these scholars were also considered during this study. The comments made by each expert regarding the identified issues and their suggestions are summarised in Table 2.

Issues 1, 2, 6, 7 and 8 were not disputed by any of the experts who agreed with the majority (4) of mainstream dictionaries. While Issue 3 had no consensus solution, all experts agreed that there should be a single well-specified standard. Issue 4 too had no clear consensus except for the recommendation that 'vOl' should be written as it is. There also appears to be a majority view that ‘Z’ should be treated as a ligature — with the only dissenting scholar too late arriving at a consensus in the interest of arriving at an overall consensus.

8. Procedures followed in State Institutions and Organisations

The following government organisations and institutes were selected for the purpose of identifying the different collation orders adopted by them for their regular work.

- National Library & Documentation Centre (NLDC)
  a. An explicit alphabetical order is available at NLDC
  b. The Sri Lanka National Bibliography is prepared according to this alphabetical order

- National Institute of Education (NIE)
  a. The NIE has adopted the alphabetical order given in the Sri Sumangala Sabdakoshaya.
  b. This order is followed when school text books and recommended books for school children are prepared.
  c. The specified alphabetical order for government examinations (e.g.: GCE (O/L) and GCE (A/L)) is also the same.
  d. Further recommendations of the NIE are given in a separate publication entitled Sinhala Lekhana Reethiya.

- Public Library – Colombo (PUB)
  a. The alphabetical order given in the Sinhala Encyclopedia is followed.

- Sinhala Dictionary Office (SDO)
  a. The criteria followed by the SDO is the criteria followed in the Sinhala Sabdakoshaya

- Sinhala Encyclopedia Office (SEO)
  a. – not yet responded –

- Election Commissioner’s Office (ELE)
  a. An explicit alphabetical order is available at ELE
  b. ‘anusvara’ and ‘visargaya’ comes at the end of vowels
  c. ‘hal’ sign comes at the beginning of vowels
  d. When there are two or more alternative forms available, the collation order is found according to the simplest form and the priority is given to the simplest form (issue #3)
  e. The letter ‘Z’ is considered as the conjunction of Ā and ə.
  f. The recommendations of the NIE given in the Sinhala Lekhana Reethiya are also followed by the ELE.

- Library — University of Colombo (UOC)
  a. The alphabetical order given in the Sinhala Sabdakoshaya is followed.

- Library — University of Peradeniya (PDN)
  a. The alphabetical order is the same as that used by UOC.

- Library — University of Kelaniya (KLN)
  a. – not yet responded –

- Library — University of Sri Jayewardenepura (SJP)
  a. –not yet responded –

- Library — University of Ruhuna (RHU)
  a. The alphabetical order given in the Sinhala Encyclopedia is followed.
Of the above, the NIE and Election Commissioner’s Office (ELE) deserve special attention. The ELE standard is of interest to this study because it explicitly addresses the issues at hand – Issues 1, 2, 3 and 5. In Issues 1 and 2, the ELE standard tallies with those of the majority of dictionaries and linguists. Interestingly, ELE has a definite recommendation for Issue 3, i.e. to locate all such form variations together at the rightful place of the simplest form with the simplest form preceding the other forms in decreasing order of simplicity. Finally, on Issue 5 (6, 7 and 8 also), the ELE standard concurs with that of the expert consensus.

Since the government recognises the NIE as the prime authority in setting educational standards the order recommended by them becomes of utmost importance. Some of the other reasons for attaching such importance to this recommendation include:

(a) The standard specified has been created by representative groups of scholars and linguists including many of those consulted in the present study.

(b) Generations of school teachers and students have already adopted this standard and hence it is the closest to a de facto standard.

(c) Their more recent publication, Sinhala Lekhana Reethiya, is widely used by state organisations including the Commissioner of Elections.

9. Summary Recommendations

The status of each of the issues considered in this study together with the recommended solution is presented below.

Issue 1: The dictionary survey and ratified by the expert consultation resolved this issue to the satisfaction of the authors: treat both the ‘anusvara’ and ‘visarga’ as appearing in the alphabetical order immediately after all the vowels. This is also further confirmed by the ELE and NIE standards which are in wide practical use.

Issue 2: The dictionary disparity with regard to the correct position for the ‘hal’ form was resolved by the unanimous opinion of the experts consulted that it should immediately follow the vowels but precede the ‘anusvara’ and ‘visarga’.

Issue 3: This was one of the issues on which empirical evidence was scarce. However, the openness of all the linguists for some standard and the simplicity rule recommended by some of them and clearly enshrined in the ELE standard is to order all forms of such words adjacent to each other beginning with the simplest form and increasing in complexity. This would prescribe the following order on the three common forms of the work karyalaya: تهديد, تهديد, تهديد.

Issue 4: This is the issue with the greatest degree of divergence in opinion. Three of the dictionaries and five of the linguists however concurred that ‘’ as used only to represent ابس and ابس’. The latter is represented as it is. This is in contrast to the original Sinhala UNICODE recommendation where ’א’ = ابس and ابس = ابس’. It seems prudent to adopt the majority opinion.

Issue 5: This was the single main success in the consensus-seeking process. It is thus recommended that ‘א’ should treated as the ligature of א + א’ so that it does not appear in the order thought to be implied in the UNICODE code chart.

Issue 6: This seemed to be an issue as it was introduced later to the Sinhala alphabet and the phonetic similarity of the letters א and א. There was confusion with the symbol א too. According to Sinhala Lekhana Reethiya, the book published by the NIE for Sinhala, and all the scholars it is accepted that the letter א should come at the end of the consonants.

Issue 7: This form is used merely for representation purposes. The underlying meaning of both א and ה is the same. Therefore they occupy the same collation position. In dictionaries these are given along with the main entry as spelling variations.

Issue 8: It is important to consider intra-word space when sorting is done in some domains (e.g. directories of names). However, this cannot be prescribed in the alphabet — it can only be done at the level of the particular application.

Based on the above recommendations and extensive testing done using early versions of the proposed collation sequence, a UNICODE Collation Element Table together with its weights is recommended as the explicit specification of the Sinhala alphabet for use in electronic processing of Sinhala. The documents observed at the organisation mentioned above and other relevant documents including the proposed Collation Element Table can be found at the URL http://www.ucsc.cmb.ac.lk/ltrl/public/collationDocs.html.

The proposed Collation Element Table can be found at the URL http://www.ucsc.cmb.ac.lk/ltrl/public/collationDocs.html

10. Conclusion

At the outset we pointed out that a complete and unambiguous specification of the Sinhala alphabet is an essential and urgent requirement for all kinds of electronic processing of Sinhala text. The process of study revealed five major areas unresolved as far as the Sinhala collation order was concerned and three other areas which needed clarification. We outlined a methodology of arriving at a set of well informed recommendations based on three sources: widely accepted dictionaries, the most respected Sinhala scholars and the most widely adopted official standards on collation sequence.
Using a consensus-based approach, we have successfully arrived at a unique collation sequence for the Sinhala language and expressed it explicitly using the UNICODE Collation Algorithm specification of the UNICODE Consortium. Testing of this specification for arbitrary lists of words is made possible by online tools available from International Components for UNICODE (ICU).

Acknowledgements

This work was partially supported by IDRC under the PAN Localization Project and sponsored by the ICT Agency of Sri Lanka. The authors are indebted to all Sinhala Language scholars, Publishers and other Practitioners who collaborated willingly in the work described in this study. In particular we wish to acknowledge the contribution of Mr. Harsha Wijayawardhana and Mr. Asanka Wasala as well as other colleagues in the Language Technology Research Centre of the University of Colombo School of Computing.

References


A. Ruvan Weerasinghe obtained a first degree in Mathematics and Statistics from the University of Colombo, a Masters in Computer Science in 1990 and a PhD in Natural Language Processing in 1994 (both from the University of Cardiff, UK). He then returned to the University of Colombo and served as a Senior Lecturer — first at the Department of Computer Science and then the University of Colombo School of Computing (UCSC). After being appointed Head of the Department of Intelligent Systems in 2003, he became Director of the School in May 2004. Ruvan may be contacted at arw@ucsc.cmb.ac.lk.

Dulip L. Herath is a Research Assistant at the Language Technology Research Laboratory (LTRL) at the University of Colombo School of Computing, Sri Lanka. He obtained his Bachelor of Science degree in Computer Science in 2004 from the University of Colombo and is currently reading for his MPhil in Natural Language Processing. His research interests are Natural Language Parsing, Localisation and Local Language Processing. As a member of the LTRL he is involved in localisation activities, namely defining the collation order of Sinhala, Interface Terminology Translation, and local language web content development using Unicode. Dulip may be contacted at dherath@webmail.cmb.ac.lk.

Kumudu Nuyanie Gamage obtained a B.A. (Hons) in Linguistics in 2004 from the University of Kelaniya, Sri Lanka. She joined the Language Technology Research Laboratory (LTRL) of the University of Colombo School of Computing as a Research Assistant the same year. As a member of a development team, Kumudu was involved in the theoretical side of Computational Linguistics with special reference to the Sinhala language. Presently she is working at the Department of Linguistics of the University of Kelaniya, as an Assistant Lecturer. Her Research interests are in Computational Linguistics, including Syntax and Sinhala Grammar, and the Analysis of differences between Sinhala and Japanese Languages with a special focus on Sinhala-Japanese/Japanese-Sinhala Translation. Kumudu may be contacted at kgamage@webmail.cmb.ac.lk.
Lessons Learnt in the Development of Applications for Remote Communities

By Alvin W. Yeo, Azman Bujang Masli, Siou-Chin Ong, Peter Songan, Jayapragas Gnaniah, Khairuddin Ab Hamid, Poline Bala,

Universiti Malaysia Sarawak (UNIMAS)

Abstract

In this paper, we highlight lessons learnt from our experience in the development of three applications for two small remote communities in Sarawak, a state of Malaysia, which has been provided with access to Information and Communication Technologies (ICTs) to remote communities, there are few studies that report specifically on development and its processes of specific technologies for the rural communities.

Keywords: Bridging the digital divide, eBario Project, ICT, Open Source, digital library, word processor, tourism website, community informatics, localisation.

1. Introduction

There are many projects aimed at bridging the digital divide that have been deployed all over the world — specifically in rural areas. Such projects have been implemented in the hope of bringing about the many potential benefits to these communities, in particular, to improve their social, economic and cultural well-being. Instances of these benefits include the elimination of the barriers to physical and virtual isolation, providing access to available information, and increasing opportunities to expand businesses to reach new markets. While many of these projects report on the provision of access of Information and Communication Technologies (ICTs) to remote communities, there are few studies that report specifically on development and its processes of specific technologies for the rural communities.

Thus, this paper aims to provide a description of and details on the lessons learnt from the development and processes employed in the implementation of three applications or systems. In the next section, this paper will provide details of the location of the two remote communities i.e., Bario and Long Bedian — the communities for which the applications were developed. In addition, the eBario project, which aims to bridge the digital divide, is also described. This project provides the context within which the three applications were developed. The ensuing section will then described the applications that were built — in particular, the rationale for the development, the methodology employed, and the outcomes of the implementation. The lessons learnt are detailed in the form of an examination from technological, operational, logistical, and strategic aspects.

2. Bario

Bario is located in the Kelabit Highlands, near the Kalimantan and Sarawak border (see Figure 1). It is the ‘unofficial capital’
Application Development for Remote Communities of the ‘land’ of the Kelabits, one of the 27 ethnic groups in Sarawak. Prior to the introduction of a daily flight into Bario, the Kelabits only means of communication with the closest town was by foot — climbing mountains, following mountain ridges, and crossing and re-crossing rivers and valleys for several weeks. Today, flying (which takes about an hour) to Bario, the main Kelabit centre, is the only practical way to get there.

Bario has a number of government offices, and also provides education and health services to the Bario community and surrounding villages. There are about 1,200 people living in Bario. The Bario district is occupied principally by the Kelabit (78%) (one of the smallest ethnic groups in Sarawak), with other ethnic groups including Penan, Kenyah, Iban, Bidayuh and Malays, Chinese, as well as some Indonesian immigrants. The majority are farmers (93%), planting wet rice as their main crop. About 5% of the population work in government offices, whereas about 2% operate personal businesses and trading. In addition to rice cultivation, the community also rears livestock such as buffalo, cattle, sheep, chicken and pigs. Some members of the community are also involved in hunting, fishing and forest gathering.

3. eBario: Bridging the Digital Divide

The idea of bringing the Internet to Bario was conceived as a research project to determine opportunities for social development available from the deployment of information and communication technologies (ICT) within remote communities in Sarawak. Desirable results from pilot studies in other developing countries have encouraged the team to work among those communities in Sarawak to have equal access to ICTs, specifically, the Internet which could provide significant improvements in their lives. This was included in the eBario project. Basically the goals of the eBario Project were to:

- Define the extent that contemporary ICTs can deliver sustainable human development and significant improvement to the lives of the community
- Demonstrate how significant and sustainable development can be achieved by remote communities through the innovative use of ICT

The objectives of the eBario project included to:

- Empower the Bario community to be able to employ ICTs to improve their livelihood through a people-centred/participatory approach
- Provide the Bario community and school children with access to ICTs through:
  - a computer laboratory at SMK Bario
  - a community telecentre at Bario

As part of the eBario project, numerous areas were identified as potential beneficiaries from the introduction of ICTs. These areas included education, culture, commerce, agriculture, health, community, technology, and human resource development.

...if you could successfully implement such a project in Bario, you could do so anywhere.

3.1 Why Bario?

While there were many communities in Sarawak that satisfied the criteria for choosing a rural remote location, Bario was selected because of its isolation. In addition, it has basic infrastructure (no 24-hour electricity supply, gravity-fed water) and no telecommunication service. This can be considered a real case of ‘digital divide’ and ‘digital poverty’. Lastly, the community’s readiness to participate, given that Universiti Malaysia Sarawak (UNIMAS) had conducted other research projects in the area and thus were known to the local Bario community. Because of its remoteness, the catch-phrase was that if you could successfully implement such a project in Bario, you could do so anywhere.

3.2 Benefits to Bario Community

Numerous benefits were realised in the areas of education, and commerce. With the community’s access to ICTs, there is increased computer literacy among the students, teachers and members of the community. Students from Bario are no longer disadvantaged when they go to the urban areas to continue their studies — they would be just as adept at using computers as their urban counterparts.

The community is able to communicate with the rest of the world due to the avail-
ability of telephones and Internet (via VSATs). The community, especially those involved in tourism, have taken advantage of ICTs — they are able to communicate with potential tourists directly via email, and confirm accommodation bookings online.

At the state level, the project has served to sensitise the State Government towards the potential for ICT-induced rural development. In particular, it has demonstrated the importance of ICTs to isolated communities that are denied other forms of infrastructure. The Government of Malaysia is paying increasing attention to rural development; different rural ICT programmes have been run — such as the Pusat InfoDesa, and Medan InfoDesa.

4. Long Bedian
A second remote community that was involved in the project are the Kayans, who live in another isolated remote location known as Long Bedian. Long Bedian is located in the Apoh Tutoh region of the Baram district, in the Miri Division of Sarawak (see Figure 1). The village comprises 180 houses and has a total population of 1,686 people. There are only two ways to get to Long Bedian from Miri town — either an express boat journey of seven hours followed by an hour-long drive to Long Bedian, or a four-and-a-half hour drive (by 4WD) from Miri. The village functions as a trading centre for the nearby villages, particularly for the Penan community. It also provides education and health services to the Long Bedian and Penan community.

The Long Bedian community comprises several ethnic groups — such as Kayan, Kelabit, Kenyah, Morek, and Punan. The Kayans are the biggest group in the village making up 95% of the population, while the Kelabits make up 3.9%. The remaining 1.1% of the total population in Long Bedian comprises the Kenyahs, Moreks and Punans. The Long Bedian community are all Christians.

The primary occupation in the Long Bedian community is farming (68.4%) — planting paddy, oil palm trees, pepper, and other crops. About 5.2% of the people are government servants, with the remainder involved in either small businesses or the private sector.

The next section details the systems developed for the communities in Bario and Bedian.

5. Software Applications Developed
There are three applications that will be covered in this paper, namely, the Bario Lakuh Digital Library, a Tourism Website and a Word Processor. These applications were the outcome of three sub-projects funded by the Universiti Malaysia Sarawak.

5.1 Bario Lakuh Digital Library
This digital library project was aimed at preserving a Kelabit oral tradition, i.e., the traditional Kelabit songs known as ‘lakuh’. The lakuh is a means of passing information about significant events on to the next generation, as well as depicting one’s feelings.

Thus, one of the objectives of the Bario Lakuh Digital Library (BLDL) project was to explore the cultural benefits of ICT in stimulating the production, protection and popularisation of Sarawak rural communities’ oral traditions, which constitute part of an indigenous knowledge system. This project, in line with the objectives of the eBario project, aimed to record and transcribe some of these traditional songs, particularly the lakuh songs.

There were three main phases to the project; Data collection, Lakuh Translation and Documentation, and Building the Digital Library.

Data Collection and Translation: Both audio and video recordings of the lakuh singers were carried out by the researchers. As the Kelabit women were only fluent in Kelabit (and spoke little Bahasa Melayu or English), it was essential to have a Kelabit speaker present. During these recording sessions, the singers were also interviewed. After recording the lakuh, it was transcribed and translated into English by Florence Apu, a qualified translator who is fluent in both written and spoken Kelabit as well as English. This translation was conducted in Bario.

Building the Digital Library: The next step was to digitise the audio and video recordings and store them in a digital library using the open-source Greenstone Digital Library Software (from the University of Waikato, New Zealand). This software allows more lakuh to be added into the existing library, if required.

Outcome: A prototype of the Bario Lakuh Digital Library was completed and has been published on CD-ROM. It contains nine lakuh sung by five Kelabit lakuh singers. The lakuh lyrics are available in Kelabit (with a translated version in English), as well as in audio and on a video recording (of the singer rendering the song). The background of the singers and details about the lakuh and its meaning are also provided.

Through this Digital Library, the cultural heritage can thus be preserved and the knowledge of the indigenous group can be passed down to the next generation. Linguists will be interested in the language used in the lakuh which has evolved over time; the lakuh are sung by women of the older generation, and thus use (untainted) Kelabit.

5.2 Tourism Website
The second application produced — a map-based tourism website — was developed as part of eBario to promote Bario as a tourist destination. It is believed that residents of Bario will benefit from eTourism.

The objectives in developing the map-based website were to:
- provide comprehensive information on Bario including maps
- provide information about lodges and homestays
- provide information about tourist guides and enable tourists to reserve a tourist guide in advance

This website was developed using the web-based system development life cycle; covering web page design, framework and content development. This website also included zoomable and interactive maps in Scalable Vector Graphic (SVG) format.

The website is now complete (see Figure 3 overleaf and also http://www.ebario.com). SVG was employed to provide maps of Sarawak, Kelabit Highlands, Bario Town, Pa Lungan and Pa Umur (villages in Bario). Key landmarks such as lodges, tourist attractions and government offices, such as the police station and immigration office, were also included.

Based on anecdotal evidence from visitors from Australia (on their way to Bario), the website provides the necessary information for visitors. Also, through the website, homestay owners in Bario have received emails from potential tourists enquiring about Bario. Presently, no data has been collected to determine the economic impact of the website. However, logged visits to the site show that there have been consistent numbers of visitors to the website, and not only Malaysian visitors (see Figure 3 overleaf). In eBario, the homestay owners are fully utilising the ICTs. They are using emails to contact their clients and are keen to use the Internet to promote their homestays and Bario itself.
5.3 Word Processor

In this project, the word processor which allowed interaction in English was customised to accommodate interactions in Kayan and Kelabit. This was implemented as part of a thesis to determine the efficacy of the existing software development lifecycle (SDLC); current SDLC is a Western construct and it was argued that the SDLC may need to be adapted to suit local contexts (Azman and Yeo, 2004).

Our first plan was to use the Open Source Software (OSS), OpenOffice. However, obtaining a build environment of OpenOffice in Windows became a major obstacle. Due to time constraints, we decided to work with a less complex software application, i.e., Abiword which is an Open Source word processor.

The development was conducted in four stages. In Stage 1, we achieved a build environment to create the software, which could accommodate different languages. In Stage 2, we identified the computing terms to be translated and to translate these terms into Kayan and Kelabit (localisation phase). Translators were identified to conduct the translations; approximately 3,000 terms had to be translated in total. In Stage 3, we tested whether the Kayan and Kelabit language could be added to Abiword. In doing this we focused mainly on the menus and tooltips, and it was successfully carried out. The Kayan version was more complete and was evaluated by native Kayan speakers.

The results indicate that the usage by the Kayans was similar to that experienced by first-time users of software in their own language. Also those who had previously used English word processors were able to identify the English equivalent first before looking for the Kayan word. Consequently, it was difficult to measure the functionality — as the users had to translate the Kayan commands back to English.

A word processor which can accommodate Kelabit and Kayan (see Figure 4) was achieved. However, the effort in adapting OSS in the project was underestimated. The team was not aware of difficulties and only decided to adopt the less complex word processor in the middle of the project. As OSS developers are located worldwide, Internet communication was the only way to get feedback. This involved participation in mailing lists and OSS community discussions.

<table>
<thead>
<tr>
<th>Digital Library (DL)</th>
<th>Tourism Website</th>
<th>Kayan, Kelabit Word Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>Capture indigenous oral traditions</td>
<td>Promote tourism</td>
</tr>
<tr>
<td><strong>Requirements Analysis</strong></td>
<td>Identify goals of DL</td>
<td>Identify goals and requirements of websites</td>
</tr>
<tr>
<td></td>
<td>Identify the singers</td>
<td>Identify people to interview and collect data from</td>
</tr>
<tr>
<td></td>
<td>Identifying translator</td>
<td>Website design: interface, navigation, databases, functionalities</td>
</tr>
<tr>
<td><strong>Design</strong></td>
<td>Design interfaces, navigation structure, functionalities</td>
<td>Website design: interface, navigation, databases, functionalities</td>
</tr>
<tr>
<td><strong>Implementation: Data collection</strong></td>
<td>Recording audio, video (on-site)</td>
<td>Recording and collection of information/content: interviews, information about culture &amp; tourist attractions, accommodation, photos, maps (on-site)</td>
</tr>
<tr>
<td></td>
<td>Interviews (on-site)</td>
<td>Interview with translators (on-site)</td>
</tr>
<tr>
<td></td>
<td>Transcription &amp; translation of songs (on and off-site)</td>
<td></td>
</tr>
<tr>
<td><strong>System Implementation</strong></td>
<td>Building the libraries</td>
<td>Development of website: webpages, databases, SVG maps</td>
</tr>
<tr>
<td></td>
<td>Digitisation of the songs</td>
<td>Incorporation of the different media and write-ups</td>
</tr>
<tr>
<td></td>
<td>Integrate into the DL CD-ROM produced (after evaluation)</td>
<td></td>
</tr>
<tr>
<td><strong>Application(s) employed</strong></td>
<td>University of Waikato’s Greenstone</td>
<td>ASP, MS SQL Server, JavaScript</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Testing and Evaluation</strong></td>
<td>Testing of the system with users [0]</td>
<td>System testing &amp; usability testing [0]</td>
</tr>
<tr>
<td></td>
<td>Refinement: Editing of the translation</td>
<td>Accuracy of information</td>
</tr>
<tr>
<td><strong>Maintenance and Operations</strong></td>
<td>CD-ROM distributed</td>
<td>UNIMAS currently maintains the website</td>
</tr>
<tr>
<td>Current</td>
<td></td>
<td>Community makes changes and uploads information themselves</td>
</tr>
<tr>
<td>Future</td>
<td>More lakuh to be added by local community after training provided</td>
<td></td>
</tr>
</tbody>
</table>

On-site: in Bario or Long Bedian; off-site (in Unimas); ◊ locals involved; [0] Optional involvement of locals

Table 1: Overview of the Three Applications Developed
sions. Delays occurred as these developers are mostly volunteers, which curtails their availability to answer queries. At present standard computing terminology of Kayan and Kelabit does not exist. Thus, provision of a tool in the target language may be a way the community can preserve the language.

6. Lessons Learnt from the Application Development
An overview of the three applications is provided in Table 1. The overview is organised according to generic software development phases and details key activities conducted as well as activities that relate to involvement of the local community. The lessons learnt from our involvement in the development of the applications here are by no means exhaustive, but do provide guidelines for those interested in developing applications for remote communities.

6.1 Crucial to Form Rapport with Target Community
Forming a rapport with the target community is of immense benefit to both parties; the developers will have access to information otherwise not available elsewhere, and the local community contributes to the successful completion of the application. (Referring to Table 1, there are numerous areas whereby the locals were involved). In the case of Bario Lakub Digital Library, during the data collection phase Florence Apu — a Kelabit and former English teacher — was able to identify with whom, when, where and how each interview could be conducted.

6.2 Identify a Local Champion
Where possible, the project team members should identify a local champion who provides the necessary information not only to the project team, but also to those on-site. The local champion would act as a motivator, at the grass-roots level, to get things done. In the case of the Tourism Website, John Tarawe was able to persuade the related parties to cooperate and provide the necessary information for the website.

6.3 Do Not Underestimate Logistical Problems
Travel to remote areas may impact on the scope as well as the project schedule and budget. Remoteness, long travel time and infrequent flights to such areas will increase the project duration. Also, such trips may be affected by inclement weather. For example, a flight delay in Bario due to bad weather could leave you stranded in Miri until the weather clears (which could take days).

The development of software for remote communities is not just about technology or logistical issues. It is about working with and for the people. As long as the needs of the people are taken into account, the technologies (regardless of what they are) will largely be accepted — albeit with some modifications to suit the local context.

6.5 System Development Both On- and Off-site
Given the difficulties of travel to remote areas (on-site), certain parts of the system development could be conducted off-site in order to reduce costs. Off-site implementation may be better since access to information/tools is easier than in the remote area. Similarly, usability tests can be conducted off-site if target users are available there.

6.6 Start Small
Where possible, applications to be developed/translated should be of a small, manageable size. Knowledge on the development envi-

6.7 Sustainability of Software Use
Training of the target community with the software is necessary to ensure maintenance and use of software. In the case of the word processor, besides training to use the tool, members of the community had to be trained to make minor modifications to the translations (in addition to being able to create the build environment for more involved changes).

7. Summary
In summary, the development of software for remote communities is not just about technology or logistical issues. It is about working with and for the people. As long as the needs of the people are taken into account, the technologies (regardless of what they are) will largely be accepted — albeit with some modifications to suit the local context.

Acknowledgements
The authors acknowledge the funding provided by Universiti Malaysia Sarawak for the following Fundamental Research Grant projects 03/14/365/2002 (102), 01/32/381/2003 (118), and 02/01/277/2002/15. The project team acknowledges the use of the Greenstone Software (http://www.greenstone.org) and AbiWord software (http://www.abisource.com) and would like to thank all those involved — in particular, the Bario and Long Bedian community.

Bibliography
Dr Alvin Yeo may be contacted at Alvin@fit.unimas.my
Abstract

Web services use Internet technologies to allow computer-based systems to communicate and transfer data in a way that provides a more seamless automated workflow. At OASIS work is being done to create a standard way for Web services to be used within the translation and localisation industry. The purpose of this article is to inform you about this work, what Web services are, and to outline a real-life case study showing how this technology is being put to use now. The article will deal in detail with the specification proposed by the OASIS technical committee for Translation Web Services. It will also describe use cases where this specification can be put into practise such as the project implemented by the Localisation Research Centre as part of the IGNITE project.

Keywords

Open standards, Web services, translation, localisation life cycle, standards development, OASIS, localisation, localization, globalization, globalisation, Localisation Research Centre, LRC

1. Introduction

Web services is a solution to the problem of computer systems not talking to each other. It uses Internet technologies to allow computer-based systems to communicate and transfer data in a way that provides a more seamless automated workflow. The likelihood is that Web services will be adopted by companies over the next few years to automate processes and integrate systems. Within the translation and localisation industry, however, there is work being done to create a standard way for Web services to be used. This work is being done at OASIS, which is the organisation for creating standards, particularly XML standards within the industry. The purpose of this article is to inform you about this work, what Web services is, and to outline a real-life case study showing how this technology is being put to use now.

This conference ended with a small group of people getting together to look at how they could progress with the idea of using Web services within the translation industry. The steering group that formed decided that OASIS would be the natural home. OASIS was established in 1993 and it is focussed on XML standards for the software industry. XLIFF (XML Localization Interchange File Format) was already being developed by an OASIS technical committee, and there was considerable support for this industry within OASIS. The OASIS technical committee was formed at the beginning of 2003 and its members include representatives from Oracle, Microsoft, IBM, Connect Global Solutions, thebigword, LISA, the LRC, and Lionbridge as well as individual members.

2. Translation Web Services

2.1 What are Web Services?

Before detailing the main features in the draft specification from the Translation Web Services (TWS) technical committee, we would like to give some background on Web services. The World Wide Web is a collection of interlinked documents that sits on the Internet, which is effectively a huge computer network that connects individual Web sites. To access a Web site a person sits at a computer and views pages through a Web browser – a process that can be considered as machine-to-person communication. With the advent of Web services the Internet is used for machine-to-
machine communication rather than machine-to-person communication. Protocols such as HTTP and standards such as XML and SOAP (Simple Object Access Protocol) are used in Web services to enable this machine-to-machine communication. This allows different systems to work together, allowing for more powerful functionality and automation.

A Web service might do something such as enable weather forecasts to be queried by remote computers over the Internet. To do this the weather forecasting company would have to create a Web service and allow it to be accessed. This is done using an XML document called a Web Service Definition Language (WSDL) document. The WSDL document describes the services which the client application is allowed access to and describes what parameters will be sent and received for each of these calls. A gardening enthusiast who is away a lot might want his computer to control his water sprinkler. By using Web services the computer will be able to find out when it is not raining and turn on the sprinkler. A protocol called Simple Object Access Protocol (SOAP) is used for this.

2.2 What is Translation Web Services?
Since January 2003 the Translation Web Services (TWS) technical committee has been working to create a standard way for Web services to be used in a multilingual context. It has concentrated its efforts on creating a standard relating to the communication between publisher and vendor companies. At the simplest level this will allow for translation and other work to be sent by the publisher to the vendor and, once translated, sent back. The draft specification covers the following areas:

- Service support
- Translation and request quote
- Status, notification and delivery
- Reference files
- Security

3. Translation Web Services Specification

Each service in the TWS specification provides two forms for interaction between client and vendor. These forms are request and response. For example, the client submits a retrieveServiceList request to the vendor. The vendor receives this request, processes it, and then returns a retrieveServiceList response to the client. The request and response forms of a service expect different inputs and produce different outputs but both request and response are needed for the interaction between and use of each service.

3.1 Categories of Services
The TWS specification defines five categories of methods or services, namely 'Service Support', 'Security', 'Translation & Request Quote', 'Status, Notification and Delivery' and 'Reference Files'. These categories form a guideline for the services that the TWS specification provides. Each category encapsulates one facet of the core work required for the completion of a translation job, from initial quote through to final delivery.

3.1.1 Service Support
The 'Service Support' category contains only one service, namely retrieveServiceList. This service allows the client to query the vendor on the type of localisation services they provide. When a retrieveServiceList request is made on a vendor a list of languages, service types, domain types, and MIME (Multipurpose Internet Mail Extensions) types that are supported is returned. In the case where no relationship exists between client and vendor, the retrieveServiceList is the first service evoked by the client to ensure the vendor meets the requirements for the potential translation job. The client can then use the information returned in their future interactions with the vendor.

3.1.2 Security
As with any transaction over the web, data security is an important consideration. OASIS defines a Web services security standard specification (WS-Security) which provides several methods for the securing of Web service-related transactions. The TWS specification relies on WS-Security to provide an end-to-end message level security and hence the specification recommends the use of username/password-based security over SSL.

3.1.3 Translation and Request Quote
This category details the services required to instantiate a job between a client and a vendor. Web services for translation uses a job ticket as a unique identifier for each project. This job ticket is created on the client side usually before a quote request. The job ticket consists of a project ID, a user ID and a unique job ID. This job ticket can then be used in all future interactions with the vendor's web service. Currently there are two methods of initiating a job in the 'Translation and Request Quote' category.

The first method is where the client submits a requestQuote service. The requestQuote service details the information pertaining to the translation job (word count, languages, etc.). The client retrieves the generated quote using the service retrieveQuote and chooses to accept or reject the quote. If the quote is accepted an acceptQuote service is activated; if it is not accepted the generated quote will expire after a certain time limit, defined by the vendor.

The second method is based upon the submitJob service. The submitJob service has similar inputs to the requestQuote service but it also contains the purchase order information found in the acceptQuote service used in the previous method. In using this second method it is automatically assumed that the job will be accepted. This interaction might be between two in-house systems, e.g. one system has content to be translated, and it contacts a second MT system and gets the content translated.

3.1.4 Status, Notification and Delivery
The Translation Web Services technical committee provides seven status, notification and delivery management services...
in the TWS specification. This set of services allows the client some control over the work that is being carried out by the vendor for a particular job. Using these services a client can check the status of and cancel or suspend a particular job. The success of each service request is dependant on the status of the job at the time of calling the service. For example you cannot cancel a job that has already been completed (for obvious reasons).

The retrieveActiveJobsList service returns to the client a list of all active jobs that they have with a particular vendor. An alternative to this is the retrieveFullJobsList service, which returns all jobs associated with a particular vendor irrespective of the current status.

A client can query the vendor using the retrieveJobInformation service and get a response containing all current information about a job. The status of the job can be deduced from the information received from the retrieveJobInformation service and possible changes to the project deadlines can be predicted. If a job is completed then the status of the retrieveJobInformation service response should reflect this.

If the information received back from the vendor after a retrieveJobInformation service request indicates that the job is completed, this job can then be downloaded using the retrieveJob service.

The client can choose to suspend a job temporarily at any time as long as the job status is not complete. This is done by making a suspendJob service request.

To remove this temporary suspension of a job (by submitting a suspendJob service request), the client can choose to resume the job using the resumeJob service.

A client can cancel a job using the cancelJob service if the job is currently active and not in a completed state.

### 3.1.5 Reference Files

The vast majority of localisation projects require not only the localisable content but also any reference files associated with the project. Reference files are not for translation but contain information that may help the translation process. Translation memories, style guides, or terminology references may be sent along with the translatable files. The 'Reference' category defines services to allow for this allocation of these files to a particular project.

A resource file can be assigned to any number of active jobs using the associateResource service.

To remove an association between a job and a resource file the disassociateResource service is used.

The client can review information about a resource file by invoking the retrieveResourceInformation service. This will return information about the resource file from the vendor: a list of jobs that the resource file is assigned to, the purpose of the resource file and whether or not the file has changed (been updated).

The TWS specification allows the client the functionality of uploading assets to the vendor using SOAP messages using the uploadFile service.

### 3.2 Services Supported in Current Specification

At this point there are 18 services supported by the TWS specification. As discussed in section 3.1, services can be categorised into one of five categories depending on their function in a translation process. These services can conversely be considered under the following three headings:

**Required Services** – these are services that are required for a Translation Web Services implementation and form the basis of a minimalist approach to Translation Web Services use.

**Recommended Services** – these services are recommended by the Translation Web Services technical committee to be used in an implementation of Translation Web Services together with the 'required' services.

**Optional Services** – these services are services that are only needed in some specific cases (enquiring about what services a vendor has to offer or setting up a first contact with a vendor by requesting a quote etc.). These services are not essential to the Translation Web Services process but are required for some scenarios.

<table>
<thead>
<tr>
<th>Required Services</th>
<th>Optional Services</th>
<th>Recommended Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>submitJob</td>
<td>retrieveServiceList</td>
<td>rejectJob</td>
</tr>
<tr>
<td>retrieveJobInformation</td>
<td>requestQuote</td>
<td>associateResource</td>
</tr>
<tr>
<td>retrieveJob</td>
<td>acceptQuote</td>
<td>disassociateResource</td>
</tr>
<tr>
<td>retrieveActiveJobsList</td>
<td>retrieveQuote</td>
<td>retrieveResourceInformation</td>
</tr>
<tr>
<td>suspendJob</td>
<td>retrieveFullJobsList</td>
<td>retrieveFullResourceList</td>
</tr>
<tr>
<td>resumeJob</td>
<td></td>
<td>uploadFile</td>
</tr>
<tr>
<td>cancelJob</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 1: List of services available in the TWS specification**

### 4. Technologies in Translation Web Services

#### 4.1 Simple Object Access Protocol (SOAP)

SOAP is a W3C-developed standard described as a communication protocol or a message passing system between two computers. SOAP is the specification that defines the XML format for these messages being passed. SOAP is one of three core XML-based standards that are the foundation of a Web services implementation (the others being WSDL and UDDI, see Sections 4.2 and 4.3).

#### 4.2 Web Services Description Language (WSDL)

A WSDL is an XML document that describes (a) a set of SOAP messages and (b) how these messages are exchanged. The WSDL file in a practical sense contains the information required by a client to access a service, i.e. what parameters
5. Localisation Life Cycle

Figure 1: Localisation life cycle incorporating the use of Web services for translation
The development model used for this reference implementation was loosely based on the prototyping model of software development. Initially one of the 18 services currently available in the standard was developed. From that the development incorporated one further service at a time until all were implemented. Throughout the development life cycle we reported back to the technical committee on any issues or suggestions for improvements that arose as we progressed.

Apart from the API for the usage of SOAP functionality Axis provides two command line utilities that are essential to the implementation of Web services. The 'Java2WSDL' utility takes pre-existing Java code and creates a WSDL file for that code. This utility can be used if there is some code that performs a specific function that you would like to make available as a Web service for others to use. The Translation Web Services standard has made a WSDL available, so this utility was redundant for our purposes. However, the second utility, 'WSDL2Java', creates the Java stubs (Java files that contain the code needed to use SOAP) required by:

(a) the server to write and deploy the service, and
(b) the client to access the service through its own code.

Figures 2 and 3 show this process. Firstly the Java stubs are created:

![Figure 2: Creating the Java Stubs from the WSDL](image)

Then the Java stubs are used by the client application to access the services and by the server to deploy the services.

![Figure 3: Connecting client and server (through the Java stubs) over HTTP using SOAP](image)

4.3 Universal Description, Discovery and Integration (UDDI)
UDDI is an OASIS-driven mechanism for clients to dynamically find other Web services. The UDDI protocol gives a company the ability to register their available Web services online, thus exposing them to potential clients. An UDDI registry service is a Web service that manages information about service providers, service implementations, and service metadata.

In summation, SOAP is the communication protocol for Web services, WSDL defines how the interaction occurs between the two computers, i.e. how to invoke the services and the UDDI is a mechanism for finding these services or registering one's own services.

5. Localisation Life Cycle

See page 10, Figure 1: Localisation life cycle incorporating the use of Web services for translation.

6. Use Cases

6.1 TWS Reference Implementation
A reference implementation of the Translation Web Services specification was undertaken by the Localisation Research Centre at the University of Limerick in Ireland as part of the IGNITE project. The Translation Web Services technical committee decided that it was important to have a reference implementation to see how the standard worked from a technical viewpoint.

The basic premise of Web services for translation is that a server machine will contain a pre-programmed set of methods or functions that a client machine can access using Web services technology. The two components involved in this interaction are the server machine and the client machine. The process of implementing Web services for each component is similar. Nevertheless there are some subtle but important distinctions to be made between both implementations.

The implementation platform of choice for this reference implementation is J2EE and the Java programming language. The rationale behind this decision was that the open source 'Apache Project' has a Java-based implementation of SOAP called Axis. Axis is defined as a "reliable and stable base on which to implement Java Web services"; it provides an Application Programming Interface (API) into the SOAP actions that are required for implementing the Translation Web Services standard. Apache Tomcat was chosen as the Web server on which to develop the Web services. Tomcat and Axis were developed under Apache and work very well together in a practical implementation environment.

The development model used for this reference implementation was loosely based on the prototyping model of software development. Initially one of the 18 services currently available in the standard was developed. From that the development incorporated one further service at a time until all were implemented. Throughout the development life cycle we reported back to the technical committee on any issues or suggestions for improvements that arose as we progressed.

Apart from the API for the usage of SOAP functionality Axis provides two command line utilities that are essential to the implementation of Web services. The 'Java2WSDL' utility takes pre-existing Java code and creates a WSDL file for that code. This utility can be used if there is some code that performs a specific function that you would like to make available as a Web service for others to use. The Translation Web Services standard has made a WSDL available, so this utility was redundant for our purposes. However, the second utility, 'WSDL2Java', creates the Java stubs (Java files that contain the code needed to use SOAP) required by:

(a) the server to write and deploy the service, and
(b) the client to access the service through its own code.

Figures 2 and 3 show this process. Firstly the Java stubs are created:

![Figure 2: Creating the Java Stubs from the WSDL](image)

Then the Java stubs are used by the client application to access the services and by the server to deploy the services.

![Figure 3: Connecting client and server (through the Java stubs) over HTTP using SOAP](image)
The first service that we implemented was the retrieveServiceList service. This service was chosen because there were no input parameters required for it. All that was required to invoke the service was an instance of the retrieveServiceListRequest class. The retrieveServiceList service returns "a complete list of services offered by a particular vendor. This will include the languages dealt with and services offered by a particular vendor" (Translation Web Services Specification Draft 1.0). After writing the server-side code to handle a retrieveServiceListRequest, i.e. return all of the appropriate values, the next stage was to create a simple test class that could instantiate a retrieveServiceListRequest and handle the results received back from the server in a retrieveServiceListResponse. With both classes now ready, we needed to deploy the services to the Apache Web server. The WSDL file also contains the location of the service, i.e. where it can be accessed from. Deployment is necessary to ensure the service is in the location as defined in the WSDL.

When the utility 'WSDL2Java' creates the Java stubs needed for the server-side machine, it also creates two other files that are used to deploy and 'un-deploy' the service to a Web server (Apache Tomcat). These files are called Web Service Deployment Descriptors ('deploy.wsdd' and 'undeploy.wsdd').

The next stage in the development of the implementation was to write the code for the rest of the services. While writing the code we encountered some issues with the specification (including inconsistencies between the schema and the specification document). These issues were quickly amended by the technical committee. During the process of coding we also made some suggestions to the technical committee about possible improvements to the specification and we were actively involved in applying these changes. For example, the service 'retrieveQuote' in the original specification did not return any information about the location of the actual quote. This was deemed to be an important piece of information for this service and was promptly included in the specification.

With the code for the implementation of the services now written, the initial service deployed (retrieveServiceList) was un-deployed and the full list of services was deployed to the Web server. A JSP (Java Server Pages) client interface was developed to allow for the input of the parameters required for each service and also to show the responses from the server.

6.2 Next Steps
The next step in the reference implementation will be to attach the front-end JSP interface to a back-end database system that returns some relevant information that is not pre-defined. The current implementation can be seen running live on www.electonline.org:8080/index.html. To view or download the source code used, please go to www.igniteweb.org/tws. Here you will also find instructions on how to install this implementation on your own local machine and server.

6.3 Lionbridge's use of Web services
Although the specification from the TWS technical committee is still at the draft stage, there has been some significant work done with Web services in the translation industry. Lionbridge has implemented a number of solutions based on Web services which have linked content management and other systems with Elcano, its online translation portal.

Figure 4: Lionbridge's use of Web services
Figure 4 shows a solution that was built for a large chemical company. Its translation process was time-consuming and error-prone. The process required more than 40 separate manual steps for each file to be translated. This led to a localisation process that was not cost-effective and delivery to target markets was unnecessarily delayed.

Working together with the company and its Systems Integrator, Lionbridge was able to propose a solution to standardise and automate parts of the process, reduce the manual effort required, and provide a more cost-effective and time-efficient way to translate the content stored within the TeamSite Content Management System (CMS).

The solution Lionbridge proposed to the company was to connect Interwoven’s TeamSite to Elcano™ (For further details see http://elcano.Lionbridge.com). Elcano offers a Web services interface providing simple and direct programmatic access to Elcano from any CMS or content repository. The Elcano Web services solution makes use of two language industry standards – both under the auspices of the OASIS standards organisation (www.oasis-open.net). XLIFF defines the structure of translation data, and TWS defines the communication between TeamSite and Elcano. Using these data transfer standards to connect to Elcano allows source content of any type to be posted directly into Lionbridge’s translation production process, its status to be tracked from the CMS during translation and, for completed translations to be retrieved without unnecessary manual intervention. In addition, once extracted from the CMS, Lionbridge can make use of a variety of translation productivity tools to ensure the consistency, accuracy and timeliness of the translation.

Freeway
Lionbridge recently introduced its new customer portal in April 2006 and the Elcano Web services described above will be ported to Freeway. More information is available at www.lionbridge.com.

References
Translation Web Services Technical Committee
www.oasis-open.org/committees/tc_home.php?wg_abbrev=trans-ws

XLIFF Technical Committee
www.xliff.org

OASIS
www.oasis-open.org/
Translation Web Services and XLIFF are developed within OASIS, which is a standards body for industry.

World Wide Web Consortium
www.w3.org
The World Wide Web Consortium is the standards body for the Internet.

Localisation Research Centre
www.localisation.ie
The Localisation Research Centre (LRC) is the information, educational, and research centre for the localisation community.

IGNITE project
www.igniteweb.org
IGNITE is an LRC-coordinated project that will pool together linguistic infrastructure resources and provide convenient access and a market place for them. IGNITE hosts an implementation of the Translation Web Services specification.

LISA - OSCAR
www.lisa.org/sigs/oscar/
OSCAR is LISA’s body for the development and maintenance of open standards for the language industry.

Note: If you are interested in joining the Translation Web Services technical committee please contact Peter Reynolds at peter.reynolds@lionbridge.com or Tony Jewtushenko at tony.jewtushenko@productinnovator.com

This whitepaper was written by Kevin Bargary and Peter Reynolds with additional contributions from Magnus Martikainen, Tony Jewtushenko, Andrzej Zydron and Reinhard Schaler. Kevin Bargary may be contacted at Kevin.Bargary@ul.ie and Peter Reynolds may be contacted at Peter.Reynolds@lionbridge.com
Abstract

Gains in productivity through translation memory-based text reuse are often offset by time spent in dealing with formatting glitches. This affects all players in the localisation industry, from the end client to the language vendor to the freelance translator. However, as a non-core activity for them, translators are less well prepared to deal with these hidden formatting related costs. This article looks from the translator’s viewpoint at the importance of formatting as part of the translator’s work, and at the limitations in dealing with formatting of the technologies now in use. It also shows how the development and implementation of standards within the localisation industry, XLIFF in particular, may impact on the situation, so that translators can once again deal only with text, as they did in pre-digital times.

Keywords
language Industry, localisation, localization, Open Standards, text reuse, TMX, XLIFF

1. TRANSLATORS TRANSLATE FILES, NOT TEXT

What translators receive for translating is files, not just text. Translators do not receive TXT files, but files with text plus formatting; with data that users can read plus code that machines can. Since many of the files translators receive have been formatted in Word, which we are all familiar with as the de facto standard for word processing, some may assume that formatting is transparent and has nothing to do with translation. However, the fact that translators, as computer users, do not need to ‘read’ the code to understand the text does not mean they do not need to pay attention to it. Translators who have been exposed to other formats have learnt that it pays to understand the differences between flat and binary files, and between structured formatting and inline formatting. The digital world has created both the file, an amazing advance from the days when text was composed on a typewriter, as well as specific technologies to deal with translation, principally translation memory (TM). This digital world has also raised the issue of formatting. It is argued here that gains in productivity through TM-based text reuse are often offset by time spent in dealing with formatting glitches.

That formatting is part of the translator’s job is obvious to any translator working in the localisation industry. Formatting, however, has not been given the prominence it deserves in training and professional development. There is no mention of it in the Language Engineering for Translators Curricula (LETRAC) Curriculum Modules (1999) in which many of the programmes with a focus on technology were first based. Even today those programmes tend to present formatting as something that will be taken care of by specialised computer software, TM suites or localisation tools. This is not quite the case yet. The importance of formatting, notwithstanding the technology currently available, has been repeatedly pointed out in the literature addressed to language vendors and end clients (Reynolds & Jewtushenko, 2005). There is a gap, however, in the literature addressed to the freelance and the trainee translator that this article will attempt to fill. Austermuhl (2001) hardly refers to formatting; Bowker (2002:37-39,118-119) and Somers (2003:18-19) only treat it marginally. Only Zetzsche (2003b) pays thorough attention to it, its focus being to give the freelancer the tools to deal with digital text.

To some extent, it is understandable that not much profile is given to formatting in academic settings. Text is the core issue for translators, formatting is not. Dealing with formatting, like dealing with invoicing, may be a most important activity, but it is non-core. Also, translating text is a complex activity that takes years to master. It involves weighing up alternative renditions of a meaning in the target language in order to choose the most appropriate one for the situation, in a context where there is rarely a clear-cut right or wrong answer. Dealing with formatting, on the other
hand, may be very complicated, but those who manipulate files will realise soon enough whether or not they have done the right thing. It is, however, part of the translator’s job, as current technology is not yet good at separating text from formatting (i.e. content from its container) within the file. In the age of the typewriter and before, formatting was unimportant. In the first stages of the digital age, it has become important, and it will continue to be important – at least until we reach the 100% XML scenario outlined below.

2. TRANSLATION IS NOT A CRAFT - IT IS AN INDUSTRY

Translation is no longer a craft; it is an industry. However, it is an industry which does not pay the translator – the freelancer at least – by the year or even by the hour like respectable professions such as law and medicine do, but by the word (or by the line, or by the page: by quantity). Translators work at the ‘wordface’ in the same way that miners work at the coalface, as Emma Wagner put it (Chesterman & Wagner, 2002:1), taking out ‘loads of translated words’ which is what language vendors sell, as Mark Lancaster, the head of SDL, a major language vendor and the most important developer of computer-aided translation tools, was reported to have said (in Fenstermacher, 2006). On the one hand, there is a low threshold entry point to the profession: any educated bilingual, given enough time and some mentoring, can become a translator; on the other, only those able to translate at great speed will be able to make it professionally profitable.

Most translators work within what has been loosely called the language industry or, more precisely, the localisation industry, also referred to lately as the globalisation industry, or the GILT (globalisation, internationalisation, localisation and translation) industry. This is an industry that, whatever name it uses, is based on selling lots of translated words, with quality often taken for granted, time-to-market an important constraint, and price paramount. This is an industry that, according to the latest calculations and with conservative estimates, will be worth more than 9 billion US dollars by the end of 2006 and will grow at 7.5 percent per year to be worth an estimated excess of 12 billion US dollars by 2010 (DePalma & Beninatto, 2006:4-5). Language vendors, like individual translators, are also paid by the number of translated words they deliver to the end client, with a benefit margin that can only be widened through increases in productivity. Despite efforts by the industry itself to monitor quality (the Localization Industry Standards Association (LISA) being a case in point) and initiatives such as the recent EN-15038 European Quality Standard for Translation Services, backed by the European Committee for Standardisation (Areavidillo Doval, 2005), the translation industry also has a low entry threshold and does not require a large amount of capital. So there is fierce competition, competition that shows in a consolidation process best reflected at the top end of the market in the mergers and acquisitions of Mendez by Lernaut&Hauspie, then of Lernaut&Hauspie and Berlitz by Bowne, then of Bowne by Lionbridge, a process that does not seem to have stabilised yet (DePalma & Beninatto, 2006:6).

This necessary increase in productivity, like that achieved in manufacturing two centuries ago, is based on the division of labour and on mechanisation. In the localisation industry, division of labour means virtual teams of translators working on a single project, with team members working off-shore to take advantage of lower salaries, or all through different time-zones if what matters is to speed up time-to-market. Mechanisation is achieved through the use of productivity tools: occasionally machine translation (MT), most often TM suites for the translation of running text, and localisation tools for the translation of short strings embedded in programme files. Productivity is achieved through the reuse of already translated text and of its formatting. In fact, it is likely that the savings in formatting reuse are greater than those achieved through text reuse although, surprisingly, no study has been done on this yet.

It is worth noting that the localisation industry does not translate – it localises. This involves project managers, graphic designers, software engineers and others working on tasks such as adaptation, quality assurance, desktop publishing adjustments and testing (Esselink, 1998:258-273), with the translator’s role limited to the replacing of natural language strings, a mere, perhaps, 30 percent of the total localisation load. But, yes, this does include the often tough task of respecting the formatting of those natural language strings. It is almost ironic that at the very moment when translation studies was ready to expand the meaning of translation beyond the tight equivalence model that dominated for decades, the localisation industry, the ‘market-driven translation theory’, moved in the opposite direction, restricting translation to an (internationalisation-driven) institutionally controlled equivalence (as Pym, 2004:62-65 explains), thereby giving the translators the added burden of having to go to great lengths to keep the formatting intact.

There are two ways for translators to deal with this formatting issue, and neither is (yet) completely satisfactory. One is overwriting the files, a bad idea if the translator does not have the application with which the original file was created and a working knowledge of that application. If the file that needs translation is flat, it is not always easy to separate translatable text from code; if it is binary, it won’t even be opened without the programme (and, often, the version) that created it. Overwriting is also a bad idea because it does not allow for the semi-automatic reuse of already translated text and, if the translator is lucky, the formatting as well. The other way, which makes more sense for the above reasons, is by using the afore-mentioned TM productivity tools, which translators are forced to do in most localisation projects anyway. The bad news, however, is that, despite claims to the contrary, TM tools do not solve many of these formatting challenges.

A brief look at the user lists for these tools (located at www.yahoogroups.com for most of the best known
commercial brands) will show the breadth of the formatting problems translators experience daily when using these tools – and what an advantage it is to be able to count on such quality peer help. Table 1 looks at data from the three lists with the greatest numbers of members and volume of messages for March 2006. Many more queries will have gone to the technical support section in the software developers’ web pages or to the language vendors that commissioned the job so the figures are indicative only. The number of messages does not reflect the seriousness of the matters dealt with in them; nor does it reflect on the quality of the particular product. The more ‘technical’ the job, the more likely it is that there will be more messages dealing with formatting issues. Wordfast, for instance, may have a lower percentage of formatting queries than TRADOS because translators working with Wordfast are likely to do less file-challenging work, not because their software is in any way superior to that of TRADOS.

<table>
<thead>
<tr>
<th>List</th>
<th>TW users (TRADOS)</th>
<th>Dejavu-I (DejaVu)</th>
<th>Wordfast (Wordfast)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formatting-related messages</td>
<td>107</td>
<td>305</td>
<td>73</td>
</tr>
<tr>
<td>Total number of messages</td>
<td>448</td>
<td>936</td>
<td>402</td>
</tr>
<tr>
<td>Percentage of formatting-related messages</td>
<td>24%</td>
<td>32%</td>
<td>18%</td>
</tr>
</tbody>
</table>

Table 1: Lists and number of formatting-related messages for March 2006

There needs to be better ways of measuring how much time and energy the average translator may use in dealing with formatting glitches. Direct observation of a ‘typical’ translator’s week, now more feasible via usability testing technology, should be attempted to give the research a more controlled, empirical outlook. The author’s limited experience as a freelance translator allows him to guess that such kind of research will also confirm that most savings gained through text re-use are offset by the amount of time spent on formatting matters. This article, however, will limit itself to supporting this hypothesis by just looking at the limitations of current technology.

3. CURRENT TECHNOLOGY PROMISES MORE THAN IT DELIVERS

Translators receive a job in one of four ways:

1. As files alone.
2. As files plus relevant sections of sentence and term databases.
3. As pre-translated files, with database information inserted in the document, as in pre-translated TRADOS files.
4. As files alone with access to databases hosted in servers.

No system, at present, avoids the problems translators often experience with formatting.

In theory, TM suites and localisation tools separate text from code before translation and then merge translated text with the original code, thus allowing for the reuse of formatting. Then they reuse content, by leveraging data from the databases of translated sentences and terms during the translation process. However, just as these tools don’t do automatic translation of the text (TM is not MT!), but just help translators with the repetitive stuff so they are free to concentrate on the more challenging aspects of the text, they do not automatically solve all formatting problems either.

The downside here is that dealing with formatting issues and code is a core activity of computer engineers, perhaps desktop publishers or even project managers, not of translators. Therefore, translators are thus less prepared to succeed here.

When language vendors and freelancers encounter the problems related to the reuse of text, they have to deal with formatting too. There are two reasons why they have to deal with formatting: Firstly, because TM databases are compiled in a proprietary format that does not allow fluid exchange of data with other TM databases – an exchange that is needed as soon as a translator works for a language vendor (or the language vendor for a client) that does not use the same programme. Secondly, because these sentence databases also keep inline formatting (the formatting within the flow of text, as opposed to structured formatting), and a segment with both the right text and format will get a better match than a segment with only the right text.

In fact, exchange of text alone between end clients, language vendors, freelance translators and TM suites is not that difficult. It can always be exported from the database to a spreadsheet programme, then from the spreadsheet to the new database. What is more difficult is working with text
that contains both inline formatting and metadata information. When TRADOS became a de facto standard in the industry, from the late 1990s onwards, most developers tried to solve the problem by making themselves compatible with TRADOS. Later, when the Translation Memory eXchange (TMX) standard emerged, they all claimed TMX compliance. However, the process of exchanging translation memory data is not always perfect; it was not perfect between TRADOS-compatible software, and it is not even perfect between TMX-certified products at the latest version of the standard, now level 1.4b (Zetzche, 2003a). In fact, it is developers themselves who simply aim for 'little or no loss of critical data' during the process of exchanging translation memory data (LISA, 2005b).

The problems grow as we move from the reuse of text to the reuse of formatting. At the point of importing a file into whatever translation tool is used, a filter is needed to convert the original file into a format that will be read by the translation editor. Creating these filters and maintaining them throughout the periodic upgrades of the programmes in which the files are composed means a waste of resources for developers – resources that would be better used if devoted to the core function of TM, which is improving the reusability of text.

These problems manifest themselves even further at the point of exporting the file for conversion into its native format, for several reasons:

- conversions are rarely 100 percent accurate
- files may not be well formed due to wrong handling by their creators (for instance, in Microsoft Word, using the enter key to change the line, or the space bar instead of the tab to indent)
- the translator may have pressed the wrong key in the translation editor

Then, we have to account for the possibility of bugs (in the file, in the filter, in the editor), for the difficulties of specific formats (MIF files, resources files), plus possible interferences of hardware / software running in the background.

There is also the issue of text expansion in translation, which will often require post-translation adjustments, particularly in presentation and design-oriented DTP files.

It is relatively easy for the translators and translation project managers to know how these productivity tools should behave in theory. The real test is in the ability of translators and managers to troubleshoot formatting problems as they arise. Allowances for budgeting and time are needed for that, which are likely to eat into most of the savings made through text reuse.

I have not referred here to other issues, such as, for freelancers the maintenance of databases and, for language vendors, the synchronisation of server databases so that they can be effectively used by different translators working at the same time. While time spent in maintenance will also eat into some of the savings from reuse, it is not directly related to formatting.

4. EMERGING TECHNOLOGY: OPEN STANDARDS

The problem with formatting is technical and the solution may be technical too. We have seen it emerging through open standards such as the above-mentioned TMX. It is widely accepted that standards benefit everyone – the product developers and businesses that depend on them as well as the actual users – and they have a positive impact on the overall economic cycle. After XML technology was developed, standards were achievable in the area of text reuse, as XML was designed precisely to separate, within the files, content from the container. The Localization Industry Standards Association (LISA) identified this and established in 1997 a specific body to develop text reuse standards. This body is entitled OSCAR (Open Standards for Container/Content Allowing Re-use).

TMX was the first such standard to emerge: version 1.0, for the exchange of text only, was released in December 1997; the latest version, 1.4b, was released in October 2004 and includes capabilities for exchange of formatting and metadata. All commercial TM suites claim to be compliant with at least version 1.1, while a few certified products, plus some non-certified products, claim to be compliant with version 1.4b of the standard. There are still the teething problems mentioned above: once again, current software often promises more than it delivers, but the situation is improving.

Term Base eXchange (TBX, version 1.0 released in April 2002) was then developed to cover the terminological exchange needs within the language industry and between tools – not only TM-based needs, but also MT-based needs. The Segmentation Rules eXchange (SRX, version 1.0 released in April 2004) followed, once it was realised that up to 30 percent of TMX-exchanged perfect matches could be lost between applications due to differences in segmentation. The last OSCAR standard, still in development, uses the official name of Global Information Management eXchange (GMX), also known as GILT Metrics eXchange. It deals with metrics rather than with text, and consists of three components: GMX Volume for word counting (the only one defined so far), GMX Complexity for the quantification of the complexity of translation tasks, and GMX Quality for the specifications of the quality requirements of translation tasks (LISA, 2005a).

All these OSCAR standards deal with the reuse of text, although GMX only does so indirectly. However, as already discussed, it is in the area of the reuse of formatting that more gains are to be expected from standards. In 2000, a new standard was developed. It is known as XLIFF (XML Localization Interchange File Format) and comes under the umbrella of OASIS, the Organization for the Advancement of Structured Information Standards. Version 1.1 of XLIFF became an OASIS Committee Specification in the Spring of
XLIFF was created for the exchange (OASIS would prefer to call it interchange this time) of translatable (or localisable) text between different file formats. With XLIFF, content can freely circulate through the localisation cycle with independence of what its native file format was, and independence of the TM suites or localisation tools that will be processed. The XLIFF conversion tool works by separating structural formatting into a skeleton file, then segmenting content and its inline formatting into translation units with its source and its target. These translation units can contain inside ‘alternative translation’ units, in most cases to hold data leveraged from a TM. Once translated, the XLIFF file merges back with the skeleton to reuse the formatting.

The XLIFF format does much more than simply interfacing with any other file format. It also allows each segment, the minimal discrete unit of translatable text that will then be kept in TM databases for recycling, to carry sophisticated metadata. This metadata can be used to track which version each segment originated from (it is as common for localisation projects to start translation before the final version of the source text has been completed, as it is to update a product, or to generate content from databases instead of static files), and to track which phase of the workflow the segment is going through, including data on tool used, job ID, client, translator/reviser, notes, metrics information, etc. Being an XML standard, it is also extensible and can accommodate future needs (Reynolds & Jewtushenko, 2005).

The XLIFF standard is being developed in line with the OSCAR standards referred to above: segmentation as per SRX rules, TM information so that it can be downloaded from/uploaded to TMX, and word counts based on GMX. Although translation units in the XLIFF format are bilingual only, multilingual projects can be dealt with by bundling together several files in a single document. This is fine, as translation is after all a bilingual activity, and a multilingual file would need to be divided into its bilingual components at some stage anyway.

There is a lot the localisation industry can gain from adopting XLIFF. Complicated projects may have to deal with over thirty different types of files, from EXE and DLL programming files to HTML and XML and their derivatives, to formats generated by content-oriented and design-oriented DTP programs, to the different Microsoft Office applications. Once this standard is adopted, instead of having to build one filter for each file format plus filters to handle data between TM suites, software developers will need just one filter for each file format. Indeed, the software that generated the files should produce this filter, thus allowing developers to shift resources to refine the algorithms so that translated text can be reused more thoroughly and easily.

In the current environment, the more that end clients rely on outsourcing localisation to multilingual vendors (MLV) and the less they spend on in-house localisation resources, the more like a ‘black box’ a whole project looks to them. With current practices end clients pass content and code on to the vendors, and later receive from the vendors the translated files ready to be imported into their document management system. In an XLIFF environment, clients will have much more control over the whole process, passing only translatable text and keeping the code (which may be sensitive in some cases) in-house. They will gain much more control of their linguistic assets also, merely by updating their own TM in the process of converting the XLIFF files to their native format. Just as importantly, they will not risk locking themselves in to a particular vendor or locking in their linguistic assets in to a particular tool.

For language vendors – particularly those at the top (MLVs) – the success of XLIFF as a standard will mean savings on management, engineering and DTP costs, without having to also lock their linguistic assets in to a particular TM tool. Their current role, which is central in the localisation process, involves dealing with all the formatting complexities the end clients do not want to spend resources on and that the freelance translators do not have the expertise to deal with. It is likely that this role will be transformed into a mere consulting job. SLVs will still retain their important role as language experts, dealing with the linguistic quality assurance of the project.

For freelancers, the success of XLIFF will mean that they will finally be able to concentrate again on text, which is their core activity, rather than on formatting, which is not. It will mean combining the advantages of the pre-digital era, when all they had to worry about was text, with those of the digital era: taking advantage of sophisticated software to reuse translated and repetitive text, allowing them to focus just on the new linguistic challenges arising. No more risks of locking themselves in to a particular tool or out from any third party information; no more need to buy several tools for different vendors: any single XLIFF-compliant tool will be enough.

5. THE 100 PERCENT XML SCENARIO

XLIFF may be the next big thing for the localisation industry, as significant as Unicode, which allowed for the easy management of character sets in any language, in any computer and with any (compliant) programme. What Unicode did for multilingual writing, XLIFF may do for transporting this written text across languages, localisation agents, software and hardware. The latest development of the past few years of moving client and localisation vendor TM databases from the desktop to the server and triggering the whole localisation process from the client’s content management system will be greatly helped by the adoption of XLIFF.

XLIFF, however, is not likely to succeed overnight. At the moment, rather than making all other formats and filters
already embracing XLIFF, although we are still at the first proprietary standards. Innovators and early adopters are JUNE 2006 successful software developers could handle it in a totally independent way. Some MLVs (gatekeepers as they are sometimes known) may resist it as it makes almost obsolete what is now a big chunk of their core activity. Like all standards, XLIFF has been developed by big players – with Novell, Oracle, Sun, and Berlitz involved first, then joined by Lotus/IBM, Moravia IT, RWS Group, and Lionbridge – but that does not guarantee its success.

On the other hand, XLIFF seems to be making inroads into the industry. Leading commercial localisation tools (Catalyst, PASSOLO, WinRC) and TM suites (SDLX/TRADOS, Heartsome) have adopted it. There is interest in the open source community in the use of this standard (Frimannsson & Hogan, 2005), with KBabel and Language Tools also offering free XLIFF-compliant tools. In some cases (SDLX for instance) the XLIFF format will interface with the native file format via its own proprietary ITD format. Heartsome, on the contrary, works directly on XLIFF, TMX and TBX standards without using any proprietary standards. Innovators and early adopters are already embracing XLIFF, although we are still at the first stages of the S-curve. For clients and language vendors, there is no longer any comparative advantage in adopting TM as most are already using it – so rather than TM being an advantage it is a necessity. However, there may be a comparative advantage in adopting XLIFF, and server TM and document management software now, before the majority does (Project-Open, 2005).

Indeed, it is easy to imagine a 100 percent XML scenario in which a more developed XLIFF specification would be able to carry out the management of information of the global enterprise seamlessly – from the authoring of text to its localisation, publishing and archiving – with processes triggered and pushed through the corresponding workflow (semi) automatically by content management software, all overseen by the project manager. Technical writers will create content on structured language and, with the help of authoring tools, through the single sourcing cycle, allowing for text chunks to be reused in other documents and to be outputted in different formats: HTML, PDF, Help, etc. (Rockley, 2002). Then, translators will move the content through the localisation cycle while reusing previously translated sentences and terms. Both technical writers and translators – language specialists in their own right – will deal only with text and, when relevant, its recycling, leaving formatting to DTP and engineering specialists who will handle it in a totally independent way.

In this scenario, successful software developers could actually afford the resources to enhance text reuse algorithms that incorporate linguistic knowledge (inflections, synonyms…) and perhaps create a new kind of language-specific TM which is more efficient for the particular language combination. Doing this would blur the distance between MT and software development, but still leave translators in charge. The process could then rightly be considered as machine assisted human translation rather than human assisted machine translation to use Hutchins’ (1992) parlance. For content creation, translation, and translation management some developers may find it useful to pursue Zydron’s (2003) ‘text memory’ xml:tm idea. Others may be interested in advancing diagnostic tools to determine whether a document should be translated by MT, by TM or without them, as the TransRouter project (Cleary & Schäler, 2000) was aiming at.

With this 100 percent XML scenario, just as in pre-digital times, translators will need to focus only on text, which is complex enough, without being distracted by the complications of formatting. After all, there will be enough challenges for freelancers in coping with the demands of translating following the imminent introduction of Web 2.0 – the Semantic Web in which machines, rather than merely displaying data as they do now, will be better able to ‘understand’ it as well (Berners-Lee, Hendler, & Lassila, 2001). Even if XLIFF succeeds, the digital world will still stir the translation profession for years.

References


References (continued)


Dr Ignacio Garcia is a senior lecturer at the School of Languages and Linguistics at the University of Western Sydney, where he has been teaching in Spanish and Translation since 1995. He is currently researching translation and new technologies, using computer-aided qualitative data analysis software (CAQDAS) on professional discussion lists to investigate how translators are responding to the challenges posed by translation memory software. He may be contacted at I.Garcia@uws.edu.au
Abstract

Microsoft is bridging the 'language divide' for speakers of minority and regional languages. Thanks to the Local Language Program (LLP), people who once were alienated by the standardisation of ICT around a handful of languages can now experience Windows and Office in their preferred language. To date, there are over 45 Language Interface Packs (LIPs) for Windows XP and 34 LIPs for Office. Examples include a version in Basque – considered to be the oldest European language spoken today – and, most recently Pashto, Luxemburgish, Nepali and Kiswahili. This initiative is a long term commitment from Microsoft and it does not end with the release of a product. The next wave of our flagship products (Windows Vista and Office 2007) will have additional language coverage with particular focus on Africa.

The Microsoft Local Language Program (LLP) is far more than an exercise in translation. Microsoft works directly with governments, local officials and language authorities to create glossaries of technology-related terms, each consisting of about 3,500 words such as 'click', 'mouse', 'file' and 'icon'. Once created, these community glossaries are put in the public domain for anyone to use, if they are not for profit.

The Microsoft Local Language Program was designed to provide speakers of lesser-used languages with an entry to information technology in a language that is familiar and which honours their linguistic and cultural traditions. For years, Microsoft has worked with governments and communities around the world to help everyone access technology. To that end, the Local Language Program (LLP) was created to bolster the local IT economy, increase individual productivity, and stimulate activity in local businesses and governments. The Program provides all people with native language interfaces that broaden the avenues to technology, reverse local 'brain drain', and preserve and honour local culture.

1. Goals of the Local Language Program

Some of the primary goals of the Local Language Program are:

1. Accelerating digital inclusion for people traditionally excluded due to the lack of available software in their native language.
2. Preserving local language and culture by using technology to drive a positive impact.
3. Extending the number of languages localised at Microsoft to include both niche languages like the lesser used Inuktitut language, spoken by the Inuit or the Eskimos of Canada, who amount to less than 30,000 speakers, to a widely used language like Bahasa Indonesia, which is spoken by 260 million speakers.

Local and regional governments around the world face great challenges building a strong information technology (IT) industry in today's global economy. Microsoft is committed to providing the tools and technologies required to develop, enhance, and expand local IT economies and enable language groups of all sizes to participate in this growth. The Local Language Program is designed to provide these opportunities to people of all regions, cultures, and languages in close collaboration with regional and local governments and universities.

The LLP addresses several critical goals of national and regional governments:

1. Enabling the use of regional languages in technology to preserve language/culture.
2. Developing a robust local IT economy.
3. Building universal IT solutions for Multilanguage cultures.

2. Three Pillars of the Local Language Program

The Local Language Program is a global initiative consisting of three 'pillars'.

2.1 Language Pillar

The foundation for the Language/Culture Pillar is the Community Glossary. The Community Glossary is a translation of standard IT terminology and is developed through voluntary collaboration between local governments, universities, and other language communities worldwide. Using the Microsoft-provided Web tools, volunteer translators and the sponsoring groups create and standardise the technical terminology glossary for their own language. Standardising the technical terminology is an important first step in developing the IT industry.

Using a volunteer process to build the glossary supports the following goals:
- Building a community and allowing the community to create the IT terminology for its local language
- Helping local groups preserve and promote their languages
- Allowing volunteers to develop their resumes and portfolios

Each new Community Glossary Web project is led by a project moderator – a volunteer with strong linguistic skills who is usually engaged by the local government. Volunteer translators join the project, suggest translations, and add comments to defend their choice of terminology. At the scheduled time, the project moderator reviews all suggestions for each term’s translation, selects the most appropriate translation, and locks each term until the glossary is complete. The local glossary project team sets the schedule, use, and distribution of the completed glossary.

### 2.2 Technology Pillar

The Technology Pillar’s focus is realised through the Language Interface Pack (LIP). Designed for regional markets that currently do not have localised versions of Microsoft Windows® and Office, the Local Language Program enables computer users to adapt their software to display many commonly used features in their native language.

Each new LIP is built using the glossary created by the Community Glossary Project in cooperation with local government, academia, and local linguistic experts.

When complete, the new Language Interface Pack is available as a free download from the Microsoft Download Center and is easily installed on a licenced copy of Windows XP and Office 2003. The Language Interface Pack also may be distributed, in agreement with Microsoft, by participating third parties such as governments, local language authorities, and universities.

### 2.3 Economy/Community Pillar

The Economy/Community Pillar is based on sparking the local IT industry into building on the momentum achieved during the production of the LIP. This has the effect of enhancing the local economy and community. As LIPs enable local IT industry infrastructure to further develop, opportunities are created to increase the value of an individual LIP, through the development of tools like spelling checkers, translation dictionaries, screen savers, and others to meet the specific demands of a burgeoning local market.

### 3. Irish Language Program

Microsoft and Foras na Gaeilge (the body responsible for the promotion of the Irish language throughout the whole island of Ireland) undertook the full localisation, development and translation into Irish of over 600,000 terms that are present across both Windows XP and Office 2003. Other partners involved in supporting the work were EGTeo, Dublin City University (DCU), eTeams (an Irish provider of translation services to businesses, governments and private clients) and National University of Ireland (NUI) Maynooth. Both packs are available to download from www.microsoft.com/ireland and www.gaeilge.ie at no charge to users.

The Irish-speaking community supported the work by contributing to the terms used in the Community Glossary, a key foundational reference for the remainder of the work. Under the auspices of the Terminology Committee in Foras na Gaeilge, the Community Glossary initiative afforded the Irish-speaking community the opportunity to make comments or contributions on the use of technical terminology. The Irish LIP is the only LIP programme delivered so far which included such extensive community engagement.

#### 3.1 Benefits for Microsoft

- Created a wonderful opportunity for staff to get involved in a specific project that delivered real value to the local community
- Enabled Microsoft to build valuable links with the Irish-speaking business and academic community nationwide (and abroad)
- Over 3,000 downloads and 4,000 copies of the programme distributed to date
- Microsoft has been nominated for the Aisling Awards for its work and awarded the prize for the company which best promotes the use of the Irish language by the annual Glór na nGael awards

"The Irish language versions of Windows and Office have generated goodwill for Microsoft that far outweighs the cost and complexity of the project," commented Tom Murphy, PR and Community Affairs, Microsoft Ireland. "The project has created a lot of excitement internally and the consistently positive feedback that we receive from Irish speakers, not to mention the awards, have been an unexpected bonus."

Speaking at the launch Mr Joe Macri, Managing Director, Microsoft Ireland added, "At Microsoft, we are committed to helping people and communities realise their potential by increasing access to technology. A group of employees at Sandyford recognised the impact that the availability of Irish products could have on communities around the country and they have driven this project with both personal and professional passion to a very successful conclusion. I am very proud of the work that has been done and am also proud of the fact that the Irish language versions of these products are being launched in 2005 – the year that we are celebrating 20 years investing in Ireland.

#### 3.2 Benefits for the Stakeholders

Four key groups can now access Microsoft Office and Windows bilingually allowing them to work, live and learn in the language of their choice:

- Public Sector – where there is a need to produce bilingual documents and content
- Schools – both Irish-speaking and non-Irish speaking; a CD copy of the Irish LIPs were distributed free of charge to all schools
- The Irish-speaking community
- Irish-speaking businesses

"The project for Foras na Gaeilge typifies the kind of language pioneering work that we see as a core role for our organisation. Through collaboration with industry and academic partners our vision is to drive the demand for use of Irish in a modern context by Irish people living their daily lives," said Seosamh MacDonncha, CEO, Foras na Gaeilge.
Speaking at the launch the Minister for Community, Rural and Gaeltacht Affairs, Mr Éamon Ó Cuív TD, said, "Under the OfficialLanguages Act we made a commitment to ensure better availability and higher standards of public services through Irish. This initiative will aid that endeavour immeasurably making it easier for our public servants to work through Irish. I must also commend the partners for their vision and commitment to a programme which will empower the educational sector, businesses and the IT community alike to communicate through our national language. This development coming so soon after the recognition of Irish as an official EU language is a very welcome development."

Mr Seosamh MacDonncha, Chief Executive, Foras na Gaeilge said, "Foras na Gaeilge continues to focus on harnessing opportunities through technology to create new tools for the Irish Language by working with major partners such as Microsoft. Foras na Gaeilge acknowledges Microsoft's commitment to Ireland which has been profoundly underscored by this project. This project also demonstrates, in a very tangible way, Microsoft as a strong corporate citizen in Ireland – both North and South. The project for Foras na Gaeilge typifies the kind of language pioneering work that we see as a core role for our organisation. Through collaboration with industry and academic partners our vision is to drive the demand for use of Irish in a modern context by Irish people living their daily lives".

Figures 1 and 2 are screen shots from Windows and Office in Irish to demonstrate what has been achieved.

4. Worldwide Language Program

There are numerous opportunities for Microsoft to use the LLP in tandem with other Microsoft citizenship efforts, such as Unlimited Potential and Partners in Learning. Through Unlimited Potential, Microsoft provides technical skills training to underprivileged individuals and communities. The Partners in Learning programme aims to empower schools to increase student learning, through teacher development and leadership.

5. Customer Feedback

What follows is a sample of the customer testimony received as Microsoft brings the Local Language Program to countries throughout the world:

5.1 Bahasa Melayu

"The availability of Windows in Bahasa Melayu – the most widely used language in Malaysia and in many other parts of Southeast Asia – will accelerate IT literacy among the Malay-speaking community and help to bridge the digital divide. Having a Bahasa Melayu interface will empower many more of our people, allowing them to become comfortable with and productive in using technology," said Dato’ Hj. A. Aziz Deraman, Director General, Dewan Bahasa dan Pustaka, Malaysia.

5.2 Kiswahili

"Localisation into Kiswahili has the following benefits: For the first time, Kiswahili speakers will be able to learn how to use a computer in their own language. In Tanzania, where Kiswahili is the medium of instruction at the Primary School level, localization is even more important. From now on, computer classes may be introduced in primary schools without any problem. In addition, localisation benefits the language by adding to its lexical stock words that were not there before, thus adding a completely new semantic domain, that of information technology. Cheers!" Kulikoyela Kahigi, Ph.D., Associate Professor, Department of Kiswahili, University of Dar es Salaam, Tanzania.

5.3 Māori

"The Māori LIP will provide opportunities for Māori language users to work in a total immersion cyber environment. The initiative completes the final step in creating a new cutting edge domain for the growth of Māori language in the public, tertiary and private sectors. It will enable literary and language scholars to work within a single language paradigm producing a new genre of students and teachers who think and write creatively in Māori. This will have enormous benefits for linguistics in Aotearoa and add value for all New Zealanders who will be able to support,
utilise and access successful language initiatives online. This will establish a new environment and a new wave of language users which in turn will add further momentum to the indigenous knowledge wave which is building in Aotearoa/New Zealand. Microsoft New Zealand has addressed a significant need in Māori language regeneration by establishing the Māori language as the world’s forty-second language to have Microsoft windows and office translated. In some ways Microsoft have given the language much more status than it has previously enjoyed in New Zealand and for this we are extremely grateful to Microsoft for their support.” Haami Te Taura Whiri te Reo Māori, chief executive of Te Taura Whiri i te Reo Māori.

5.4 Welsh

"To summarise, the feedback has been phenomenal, fantastic and great. More please! It's the best project I've ever worked on, and the enquiries are still coming in, seven months and more after launch!" Jeremy Evas, Leader of the Research and Grants Unit of the Welsh Language Board.

6. Summary

Microsoft recognises that local and regional governments are facing many challenges as they work to build strong local IT economies. The Local Language Program demonstrates the company's commitment to providing the software products and tools necessary for success in economic growth and language preservation. With the Microsoft Local Language Program, PC users around the world can bridge the digital and language divide, thereby helping them to reach out to new markets, create customised local solutions, stimulate the development of their local IT economy, and preserve and honour their culture. You can learn more about the Local Language Program by contacting your local Microsoft Sales office or by visiting www.microsoft.com/industry/government/locallanguage.mspx.

A list of Microsoft Sales offices is available at www.microsoft.com/worldwide.

© 2006 Microsoft Corporation. All rights reserved. Microsoft, Excel, Outlook, PowerPoint, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Liam Cronin has worked in Microsoft EPDC since September 1991, in a combination of products from Excel 3.0, Windows 3.1, Windows for Workgroups 3.11, Windows 95, Money, MSN Money, Works and Digital Imaging. He is currently Group manager at Microsoft Ireland, looking after program management, user assistance, testing and engineering for Consumer Software titles. Liam can be contacted at liamc@microsoft.com or +353-1-7064989.

Table 1 details a comprehensive list of languages covered for Windows and Office:

<table>
<thead>
<tr>
<th>Language</th>
<th>Windows XP LIP</th>
<th>Office 2003 LIP</th>
<th>Office 2003 Full</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afrikaans</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Albanian</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Armenian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Azeri</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basque</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Bengali</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bosnian Cyrillic</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Bosnian Latin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bulgarian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Catalan</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Croatian</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Estonian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Filipino</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Galician</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Georgian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gujarati</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hindi</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Icelandic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indonesian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inuktitut</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Irish</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>isiZulu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kannada</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kazakh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kiswahili</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Konkani</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latvian</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Lithuanian</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Beavers, Maple Leaves and Maple Trees

A Study of National Symbols on Localised and Domestic Websites

Abstract

Because a national symbol appeals to the sense of collective identity shared by the members of a nation, its use in localised websites by companies from outside the nation merits reflection. In this paper, a case study of thirty of the largest American corporations is used to explore how common it is for national symbols to be incorporated into websites localised for Canadian users. The results are then compared to the use of national symbols on the websites of thirty of the largest Canadian corporations to determine whether national symbols are adopted more frequently by domestic or international companies. The paper ends with some reflections on the inclusion of national symbols within a localised website and the ambiguity of their meaning.

Keywords

localisation, collective identity, symbolism, national identity, website translation, adaptation, national symbols

Introduction

Given that the purpose of localisation is to ensure that localised products are adapted to the conventions of a given locale (Lommel 2003, p.5), a successfully localised website should not immediately strike targeted users as being different from a site designed by a domestic company. As part of the process of adapting the site for a new locale, especially when targeting an entire country, localisers may decide to incorporate images of locale-specific national symbols such as flags or monuments. Singh and Pereira (2005), for instance, recommend including symbols and "pictures of national identity" such as those that emphasise architectural achievements or national pride on websites designed for collectivist locales (2005, p.83). They stress that domestic companies will be incorporating such symbols in their own websites and advertisements, implying that international companies will be at a disadvantage if they do not follow suit.

Yet this assumption may merit further reflection. This paper will raise questions about the use of national symbols on websites and explore the issues surrounding the inclusion of national symbols within localised sites. It will do this by first exploring what national symbols are supposed to represent and then studying examples of Canadian symbols used in localised and domestic websites.

1. National Symbols

National symbols help form and maintain national identity, mark a nation's collective memory, preserve its shared past and represent the power of a state to define a nation (Geisler 2005, pp.xv–xvii). In a way, national symbols act much like the logo of a corporation, as they are a means by which the State can depict an image of itself to members and outsiders alike. Much like a corporate logo, national symbols represent a nation's key values and are chosen because they have special significance for the nation and its members. As Smith (1991, p.77) notes, national symbols, customs and ceremonies make the concepts of a nation visible for all members and appeal to their emotions.

However, what exactly constitutes a national symbol is not unanimously agreed upon. Smith (1991, ibid.), for instance, groups symbols, customs and ceremonies together and considers flags, anthems, parades, coins, capital cities, folk costumes, folklore museums, war monuments, passports and borders to be "obvious" examples. Cerulo (1995, p.13) adds mottos and shrines to this main list, and Smith later expands his initial examples with a series of "hidden" ones, including popular heroes or heroines, fairy tales, educational practices and military codes. He asserts that these symbols, customs and ceremonies are the ways of acting shared by a "community of historical culture" (1991, p.77). Geisler, on the other hand, argues that Smith's typology may be too broad. He suggests that a narrow typology of important national symbols would minimally include the flag, anthem, national holidays, currency, capital and major national monuments, with the flag being the most important and the others ranked somewhere below it (2005, pp.xxi – xxii).

These symbols, Geisler asserts, serve to support and

1 The author would like to thank Clara Foz for her feedback on an earlier version of this paper.
reinforce a nation’s identity, both within its borders and to the outside world. Each time such a symbol is “actualised” – whenever an anthem is sung or a flag is raised, for instance – it reminds members of the nation that they share a common past and are bound by a collective identity. In fact, only through constant repetition of a symbol in the media, political speeches, public ceremonies, etc., do members of a nation become attached to it (Geisler 2005, pp. xix, xxvii). Similarly, David Bell (2001, p.95, following Anderson 1983) suggests that nations are imagined communities and that symbols are essential for this community to become a nation, since members can interact with others only through shared ‘things’ such as an anthem or flag and a set of customs and rituals.  

Because national symbols depict a nation’s history, values and identity, they send an ambiguous message to users when they are incorporated into a localised website. On one hand, the symbols act as a sort of logo that identifies the site as the Canadian version. A Canadian flag beside a “change location” link may simply serve to distinguish this site from one that has, say, a French, Japanese, or Chilean flag, alerting users that they are indeed browsing the site designed for their locale. The ambiguity results from the fact that the “imagined community” described by Bell and Anderson is also projected by localised websites incorporating local symbols. Just as a logo stamped on a product’s packaging signals to consumers that the product has been made by and comes from a given company, so do national symbols signal that a website has been made in and is part of a given nation. Users may therefore interpret a national symbol to mean that both the company and the user belong to the same imagined community, share the same collective identity, and are bound by the same common past.

2. Use of Canadian Symbols

The Government of Canada officially recognises three national symbols in addition to the national flag, colours, seal, and anthem: the beaver, maple tree and maple leaf (Government of Canada, Canadian Heritage 2004). All of these symbols are infused with special historical significance. The beaver, which became "a symbol of the sovereignty of Canada" when the National Symbol of Canada Act was passed in 1975, is a reminder of the importance of the fur trade to the early Canadian economy in the 17th, 18th and 19th centuries, when thousands of Canadian beaver pelts were shipped to Europe annually for use in fur hats. It appears on the Canadian five-cent coin and was featured on the first Canadian stamp. The maple tree, officially recognised as Canada’s "arboreal emblem" in 1996, "played a meaningful role in the historical development of Canada and continue[s] to be of commercial, environmental and aesthetic importance to all Canadians" (ibid). At least one of the ten species of maple native to Canada grows in every province, and the sap is used to make maple syrup, of which Canada is the leading global producer. Finally, the maple leaf (Figure 1), incorporated into the Canadian and Ontario flags in 1965, appears on the one-cent coin and is featured in The Maple Leaf Forever, a song composed for Canada’s confederation in 1867 and an unofficial English-Canadian anthem for several decades.

Figure 1: The 11-point maple leaf, an official symbol of Canada

Though these symbols may be accorded official State-recognised status, several others could be considered to have semi-official status, even if one follows only the narrow typology of symbols offered by Geisler (2005). One could reasonably include the moose, found on the twenty-five cent piece; the loon, depicted on the one-dollar coin; the polar bear, which appears on the two-dollar coin; and the Bluenose, a fishing schooner built in the 1920s that was renowned for winning several international races during the 1920s, represented Canada at the 1933 Chicago World Fair, was sent to England on behalf of Canada in 1935 for the Silver Jubilee of King George and Queen Mary, and is featured on the ten-cent coin (Province of Nova Scotia, Department of Tourism, Culture & Heritage 2004). And since Geisler also includes major national monuments, one might add to this semi-official list the Canadian Parliament buildings or the War Memorial in Ottawa, where Remembrance Day ceremonies are held each year.

Other regions of Canada have their own provincial or ‘national’ symbols as well. All of the country’s ten provinces and three territories have an official coat of arms, flag and flower and many regional groups, such as Franco-Ontarians, also have officially recognised flags or emblems. While such symbols could also be used by localisers to target a website to a specific group of Canadians, this paper will focus only on national symbols representative of Canada as a whole rather than a particular region. A future study will encompass a wider range of symbols, as their use on a website will help indicate which particular segments of the Canadian population a company may be trying to target. For the purposes of this case study, both the official and non-official symbols mentioned in the two preceding paragraphs were considered to be national symbols of Canada.

---

2 See also Hall (1996) for further discussion of imagined communities and cultural representation.
3 Canada is responsible for approximately 85 percent of world maple syrup production. See fact sheet available at: http://ats.agr.gc.ca/supply/3310_e.htm [accessed 12 April 2006].
4 It was in fact dubbed the "Queen of the North Atlantic fishing fleet."
5 Though other provinces or territories usually use the term ‘national’ to refer to the federal government and Canada as a whole, Quebec often uses the term to refer to Quebec institutions and symbols. Thus, Quebec’s provincial legislature is referred to as the Assemblée nationale or National Assembly, the Quebec government’s highest award of distinction is the Ordre national du Québec, the region surrounding Quebec’s provincial capital is referred to as the capitale-nationale and Saint-Jean-Baptiste Day, an official holiday only in Quebec, is referred to as the Fête nationale.
2.1 Case Study: Methodology

To study the use of Canadian national symbols on localised websites and those of Canadian companies, thirty of the largest American corporations and thirty of the largest Canadian were compared. A Canadian company has been defined as one that has its headquarters in Canada and is not a subsidiary of an international company. Sears Canada, for instance, would be considered Canadian even though Sears Roebuck owns more than 50% of its shares.6

The American companies were selected based on the Fortune 500 list published by Fortune magazine on 18 April 2005, while Canadian companies were selected based on the 2005 Top 1000 Companies rankings compiled by Report on Business Magazine, which is published by The Globe and Mail, a major Canadian daily newspaper.

The Fortune and Globe and Mail rankings were chosen for two reasons. First, given the fact that localisation involves a considerable investment of financial and human resources, larger corporations are more likely than smaller companies to have international operations and localised websites for foreign markets. In addition, the corporations that head the list fall under various industries, making the sample more representative of large Canadian and American corporations in general rather than of those in a particular sector. Though energy companies do figure prominently in both lists, Fortune's top fifty also includes department stores such as Wal-Mart and Costco, specialty stores such as Home Depot, and manufacturers such as Procter & Gamble, Ford, and Dell, while the Globe and Mail top fifty includes banks such as CIBC, RBC, BMO and TD, grocery retailers such as Loblaw, telephone utilities such as Bell Canada, and manufacturers such as Magna International.

The American companies were selected from the top fifty-three of the Fortune 500, beginning with Wal-Mart (#1) and ending with Merrill Lynch (#53). In order of ranking, the website of each corporation was checked, and if a Canadian version of the site was available, the company was included in the study. Twenty-three of the top fifty-three companies had to be excluded as no Canadian version of their website was available. In each case, the next-ranked company was chosen so that a total of thirty could be included in the case study. Websites were considered localised for Canada when the US parent company had a global gateway from which a Canada or Canadian site could be accessed or when a link to the Canadian version was posted on the American website. When English- and French-Canadian sites were available, both versions were consulted; otherwise, the English-Canadian site was considered to be the localised version.9 In total, thirty-seven websites representing the thirty companies and their subsidiaries were consulted.

As a point of comparison, thirty Canadian companies were selected from among the first thirty-nine on the ROB Top 1000 Companies list. For the purposes of this study, when both a holding/parent corporation and its subsidiaries were listed, they were not counted as separate companies, though the websites of both the parent and the subsidiary were consulted. For instance, Power Corp (#26) holds Power Financial Corp (#13), which in turn holds Great West Life (#20). All three corporations were counted as only one of the thirty in this case study, though the website of each was examined. In addition, whenever an additional Canadian version of a website was separate from the main corporate site, both were consulted (e.g. www.loblaw.com, the corporate website for Loblaw Companies Limited, was consulted, as were the websites of its retail locations and brands, including Loblaws, No Frills, and President's Choice). In total, fifty-two websites were consulted, representing thirty companies and their subsidiaries. These websites were considered 'domestic' in contrast to the 'localised' sites of the American corporations.

---

6 The companies listed in the ROB report are all publicly traded on the Toronto Stock Exchange. The problems that arise from this definition will be explored in the next section.
7 Of the top thirty-nine companies on the Globe and Mail list, for instance, ten are classified as oil- and gas-related (oil and gas producers, integrated oil, gas pipelines, etc.).
8 e.g. Bank of America (#18), Target (#27), Morgan Stanley (#36) and Metlife (#37).
9 A small percentage of both the localised and domestic sites were available only in English: 18 of the 52 domestic and 6 of the 37 localised.
10 Manulife Financial has this same image on its homepage.
RBC and served only to link to a news article about the award and to the survey website: http://www.mostrespected.ca.

**Figure 3:** Example of a Canadian symbol included within an image on a website.

### 2.2 Case Study: Findings

Of the thirty American companies with localised websites studied for this paper, a total of twelve incorporated Canadian symbols. Maple leaves were used by six of these companies, while the Canadian flag was used by the other six. American International Group, which owns AIG Life, used a Canadian flag on the AIG website and a maple leaf on that of AIG Life. Only one site, General Motors Canada, used both a maple leaf – as part of its logo – and a flag, while General Electric included both maple leaves and a photograph of the CN Tower, arguably a Canadian national monument, as it is billed as “Canada’s wonder of the world” on the CN Tower website (www.cntower.ca). No other national symbols (e.g. beaver, Parliament) appeared to be used by any of the companies. Table 1 summarises the use of national symbols on these websites.

<table>
<thead>
<tr>
<th>Fortune Ranking</th>
<th>Company</th>
<th>Canadian website</th>
<th>Symbol(s)</th>
<th>Location of symbol(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Exxon Mobil</td>
<td><a href="http://www.exxonmobil.com/Canada-English/HR/HR_Can_Homepage.asp">http://www.exxonmobil.com/Canada-English/HR/HR_Can_Homepage.asp</a></td>
<td>Canadian flag</td>
<td>HR page (home page of Exxon Mobil Canada. The site has been partially localised.)</td>
</tr>
<tr>
<td>3</td>
<td>General Motors</td>
<td><a href="http://www.gmcanada.com">www.gmcanada.com</a></td>
<td>1. Maple leaf 2. Canadian flag</td>
<td>1. In GM Canada logo 2. In a photo GM dealership on the Site Map/About Us pages</td>
</tr>
<tr>
<td>9</td>
<td>American International Group</td>
<td><a href="http://www.aig.com/gateway/home/1-113-Canada_index.htm">http://www.aig.com/gateway/home/1-113-Canada_index.htm</a></td>
<td>Canadian flag</td>
<td>Beside ‘change location’ link on navigation bar</td>
</tr>
<tr>
<td>*</td>
<td></td>
<td><a href="http://www.aiglife.ca/">http://www.aiglife.ca/</a></td>
<td>Red maple leaf</td>
<td>On homepage, beside ‘AIG Life of Canada’</td>
</tr>
<tr>
<td>19</td>
<td>State Farm Insurance</td>
<td><a href="http://www.statefarm.ca">www.statefarm.ca</a></td>
<td>Red maple leaf</td>
<td>On homepage, beside ‘statefarm.ca’</td>
</tr>
<tr>
<td>24</td>
<td>Pfizer</td>
<td><a href="http://www.pfizer.ca">www.pfizer.ca</a></td>
<td>Red maple leaf</td>
<td>On homepage, beside ‘healthcare in Canada’ heading</td>
</tr>
<tr>
<td>28</td>
<td>Dell</td>
<td><a href="http://www.dell.ca/">http://www.dell.ca/</a></td>
<td>Canadian flag</td>
<td>Beside Dell Canada logo on navigation bar</td>
</tr>
<tr>
<td>30</td>
<td>Johnson &amp; Johnson</td>
<td><a href="http://www.jnjcanada.com">www.jnjcanada.com</a></td>
<td>Canadian flag</td>
<td>Homepage</td>
</tr>
<tr>
<td>32</td>
<td>Time Warner (AOL, Time)</td>
<td><a href="http://www.aol.ca">www.aol.ca</a></td>
<td>Red maple leaf</td>
<td>Beside search bar. Note: maple leaf not on Quebec site (<a href="http://www.aol.qc.ca">www.aol.qc.ca</a>)</td>
</tr>
</tbody>
</table>

* Table 1: National symbols used in websites localised for Canada
* Each of these companies was considered a subsidiary of the Fortune – or Globe and Mail – ranked company just above it (e.g. Bell Canada and Telesat are subsidiaries of BCE Inc.). With the exception of Bell Canada, these companies did not appear in the Fortune 500 or Globe and Mail top 1000.
<table>
<thead>
<tr>
<th>G&amp;M Ranking</th>
<th>Company</th>
<th>Website</th>
<th>Symbol(s)</th>
<th>Location of symbol(s)</th>
<th>Head Office</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Canadian Imperial Bank of Commerce</td>
<td><a href="http://www.cibc.com/ca">http://www.cibc.com/ca</a></td>
<td>Photo of CN Tower</td>
<td>CIBC world markets page</td>
<td>Toronto, Ontario</td>
</tr>
<tr>
<td>10</td>
<td>Petro-Canada</td>
<td><a href="http://www.petro-canada.ca">www.petro-canada.ca</a></td>
<td>White maple leaf</td>
<td>Used in logo</td>
<td>Calgary, Alberta</td>
</tr>
<tr>
<td>14</td>
<td>BCE Inc.</td>
<td><a href="http://www.bce.ca/">http://www.bce.ca/</a></td>
<td>-</td>
<td>-</td>
<td>Montreal, Quebec</td>
</tr>
<tr>
<td>*</td>
<td>Bell Canada (#15)</td>
<td><a href="http://www.bell.ca">www.bell.ca</a></td>
<td>-</td>
<td>-</td>
<td>Montreal, Quebec</td>
</tr>
<tr>
<td>*</td>
<td>-</td>
<td><a href="http://www.telesat.ca/">http://www.telesat.ca/</a></td>
<td>Red maple leaf</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>16</td>
<td>Cdn. Natural Resources</td>
<td><a href="http://www.cnrl.com">www.cnrl.com</a></td>
<td>White maple leaf</td>
<td>Used in logo</td>
<td>Calgary, Alberta</td>
</tr>
<tr>
<td>18</td>
<td>Canadian National Railway Co.</td>
<td><a href="http://www.cn.ca">www.cn.ca</a></td>
<td>Canadian flag</td>
<td>On the homepage, in an image of a small, red CN train filled with people. Both US and Canadian flags are flying on the train.</td>
<td>Montreal, Quebec</td>
</tr>
<tr>
<td>*</td>
<td>-</td>
<td><a href="http://www.sunoco.ca/">http://www.sunoco.ca/</a></td>
<td>Canadian flag</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>24</td>
<td>Husky Energy</td>
<td><a href="http://www.huskyenergy.ca">www.huskyenergy.ca</a></td>
<td>Maple tree branch</td>
<td>Photo on the About Husky — Health Safety &amp; Environment page</td>
<td>Calgary, Alberta</td>
</tr>
<tr>
<td>34</td>
<td>Talisman Energy</td>
<td><a href="http://www.talisman-energy.com/">www.talisman-energy.com/</a></td>
<td>Photo of a maple tree</td>
<td>On the About Us page</td>
<td>Calgary, Alberta</td>
</tr>
<tr>
<td>35</td>
<td>Enbridge Inc.</td>
<td><a href="http://www.enbridge.com">www.enbridge.com</a></td>
<td>Photo of Enbridge van with CN tower in far background</td>
<td>On the Library page</td>
<td>Calgary, Alberta</td>
</tr>
<tr>
<td>40</td>
<td>Ipsco Inc.</td>
<td><a href="http://www.ipsco.com">www.ipsco.com</a></td>
<td>Canada goose</td>
<td>Used in logo</td>
<td>Regina, Saskatchewan</td>
</tr>
</tbody>
</table>

Table 2: National symbols used in the websites of Canadian companies

* Each of these companies was considered a subsidiary of the Fortune – or Globe and Mail – ranked company just above it (e.g. Bell Canada and Telesat are subsidiaries of BCE Inc.). With the exception of Bell Canada, these companies did not appear in the Fortune 500 or Globe and Mail top 1000.
The websites of the thirty Canadian companies and their subsidiaries also included Canadian symbols, to almost the same extent: thirteen of the fifty-two sites – representing eleven of the thirty companies – had images of maple leaves or trees, the Canadian flag, a beaver11 or the CN Tower on their sites. Many of the websites in this group were the corporate sites intended both for Canadians and international users and hence would not necessarily focus on the company’s ‘Canadianness’; however, some of the .com sites included national symbols, while many of the .ca sites did not. For instance, none of the Loblaw subsidiaries or brands – including No Frills, Fortinos, Maxi, Zehrs Markets and Independent – used Canadian symbols, though each of these latter sites has been localised for users within the province(s) where that chain of grocery stores is located. This shows that a site does not have to be targeted to just Canadians for a company to highlight its Canadian roots. Table 2 summarises the use of national symbols on these websites.

As Tables 1 and 2 illustrate, Canadian symbols are used on both domestic and localised websites. In both sample groups, the maple leaf and Canadian flag were favoured over other official or semi-official symbols, and though these emblems were most commonly located on the homepages of localised sites and in the logos of domestic sites, they were also found on various other pages. The results raise intriguing questions, among which are what the national symbols are intended to represent and what their function is supposed to be. On the localised sites, one can reasonably assume that a Canadian flag – as used on the Canadian version of the AIG website, for instance – is intended to help users distinguish one locale from another and signal that the parent company is making an effort to be part of the Canadian community. This assumption is supported by the fact that of the thirty-seven localised websites in this study – including those that did not incorporate national symbols – only two, those of General Motors Canada and AIG Assurance12, did not appear to have a link to their parent company. Thus, the fact that these localised websites are part of a larger, global operation is not actively concealed from users in the targeted locale, regardless of whether or not national symbols are used.

Yet the function of national symbols is not necessarily the same in the domestic websites. Because a Canadian-owned company is actually part of the Canadian ‘imagined community’, the national symbols on its website signal not only that the site is intended for the English- and/or French-Canadian locales, but also that both the company and Canadian users share the same imagined identity. The Canadian symbols create a bond (see Cerulo 1995, p.16) between the company and the user, appealing to the latter's sense of collective identity, belonging and patriotic desire to support local businesses. The symbol may or may not achieve this effect, but it certainly performs this function.

And here lies the issue upon which further reflection is merited. Unless a user actively searches through a website to determine whether the company is in fact Canadian, how is he or she supposed to know what the national symbol is intended to represent? When an image of a maple leaf, Canadian flag, or Canadian monument is found on a home or start-up splash page of a .ca website, a user’s first inclination would be to identify the company as Canadian, whether or not this is actually the case.

The issue is made more complicated by the fact that in some ways a subsidiary of an American company is still a part of Canada, though not technically owned by Canadians13. Both GE and General Motors Canada, for instance, have long histories in Canada and employ thousands of Canadians. GE’s first manufacturing facility in Canada was opened in 1892, while General Motors Canada was established when GM bought the family-run and Canadian-owned McLaughlin Motor Car Company in 191814.

In other cases, a Canadian company, though not a subsidiary of a larger, international operation, may not be entirely Canadian-owned. As mentioned earlier, Sears Canada is not a subsidiary of Sears Roebuck15, since Sears Canada was actually formed as a 50-50 partnership between The Robert Simpson Company, a Canadian retailer, and Sears Roebuck in 1953. However, Sears Roebuck has since increased its ownership of Sears Canada: in 1984 it held 62.6% of the company, but by 1996 it held a smaller majority of 55% of shares16. As ownership changes hands over time, does a Canadian company become more or less Canadian? And if so, should its ‘right’ to use Canadian symbols be revoked? Corporate ownership is often difficult to precisely determine, which only adds to the ambiguity surrounding what national symbols really represent on commercial websites.

Even the legislation related to the use of Canadian symbols does not completely elucidate the issue. Several symbols are protected by Canadian law. The national flag and coat of arms, for instance, are protected by The Trade Marks Act, which forbids commercial use of these symbols without permission from the federal government’s Department of Canadian Heritage17. The maple leaf itself is protected by both an international treaty (Paris Convention for the
Protection of Industrial Property) and Canadian legislation (Order in Council P.C. 1965 – 1623), though the exact symbol referred to is the 11-point maple leaf (Figure 1) that appears on the Canadian flag. It is likely that because of these regulations, the maple leaves included on sites such as Procter & Gamble or Merrill Lynch are not exactly the same as the 11-point trademarked symbol protected by the Canadian government. In this way, the symbol itself can still be used in an effort to create a bond between the corporation and website users without actually infringing on trademark laws.

Yet, even if the maple leaves incorporated into a localised or domestic website are not identical to the 11-point leaf officially recognised and protected by the Canadian government, they still function in much the same way as their official counterpart. A maple leaf – and other Canadian symbols – will appeal to a user's sense of collective identity, regardless of whether or not it has eleven points and is identical to the one on the Canadian flag. And because any national symbol will operate on more than one level, no company can be sure that it will be received and interpreted as intended.

Consider, for instance, the fact that in Canada national symbols do not evoke the same reaction from all Canadians. Supporters of Quebec independence or sovereignty often view the Canadian flag and maple leaf negatively. The Parti québécois, a secessionist provincial political party, for instance, once refused federal funding for renovations to the Quebec City zoo and aquarium because the grant was tied to the condition that bilingual signs be posted and the Canadian flag fly over both buildings for forty years. This offer was decried as "une tentative de relativiser notre statut national" ["an attempt to dilute our national status"18], and the Quebec government instead funded the entire $38 million dollar project itself (Lessard 2001, p.A1; Séguin 2001, p.A01). Yet only one of the sites in this study seemed aware that the maple leaf could potentially have a negative, rather than a positive, effect on a user's reception of the local site: while the AOL English-Canadian website included a red maple leaf, this symbol was not found on the AOL French-language Quebec site, though no official Quebec symbols such as the flag or fleur-de-lys were used instead19. AOL thus differentiates between English Canadians, who are expected to be receptive to the maple leaf, and French Canadians, who may not be.

National symbols, then, risk not only being misinterpreted by users, but also, in some cases, acting contrary to the corporation's intention: instead of creating a bond between users and the company, red maple leaves and Canadian flags may actually alienate certain segments of the intended audience, who do not consider the national flag and other official emblems representative of their national identity.

Yet, using the official symbols of a particular group of Canadians (e.g. fleur-de-lys for Quebecers, especially the French speakers) to better reach a group that feels little attachment to the national symbol would simply create more problems, since an additional site would have to be created to target this locale. Instead of offering English- and French-Canadian sites, a company would ideally have to create one for English Canada, one for French Canada (since French speakers live throughout the country) and yet another for Quebec, available in at least two languages, as both English and French speakers reside within the province. Localisers would be creating largely unnecessary segmentation and additional websites simply to include various official symbols that may not even be received as intended. And the smaller the group targeted by the localised site, the less likely the company is to see a significant return on its investment.

As the results of this case study indicate, Canadian and American companies do not uniformly use national symbols on their (localised) websites. Since approximately half of both the American and Canadian companies included some officially recognised symbol, it is unclear what the symbols are supposed to represent. At best, they are used haphazardly by companies and are included or removed when sites are redesigned. No national symbols were found on the current Wal-Mart Canada site, for instance, though in 2005 a red maple leaf appeared on the home page beside the "Welcome to Wal-Mart Canada Corp." heading. Canadian website users are therefore receiving unclear messages about a company's status in Canada. They may not realise that a Canadian company without a national symbol on its site is in fact Canadian, and they may mistakenly believe an American company is Canadian owned or has its headquarters in Canada simply because it uses maple leaves, maple trees or the Canadian flag somewhere on its website.

**Conclusion**

As discussed, national symbols in localisation have a dual nature. Superficially, they are accessories used to designate the locale for which a website has been designed. But because they also have a more figurative function – that of reaffirming collective identity – they may be interpreted by some users as an indication that both the company and the user are part of the same imaginary nation represented by the symbol and thus share the same historic roots and core values.

The goal of localisation, notes Yunker (2003, p.18), is not to "trick" users into thinking a company is local, but rather to let them know that the company understands the needs and wants of users in a given locale. Are national symbols necessary for transmitting this latter message? Not necessarily. In fact, I would argue that precisely because localisation is not supposed to deceive users, a corporation...
should carefully consider how national symbols might be interpreted before deciding whether or not to include them in a localised website. Localisers would also be wise to consider the political implications of incorporating federal symbols into websites when such symbols risk alienating or, at the very least annoying, users in the locale.

National symbols are not the only way of appealing to a locale in which collective values are very strong. Focus can still be placed on the company's place in and contributions to the area by highlighting its involvement in the local community, its donations to local charities, the number of jobs it has created within the region, etc. In this way, the chance for users to misinterpret a company's intentions or origins would be reduced and fewer users would be likely to be antagonised, while the company's contributions to the locale would not be overlooked.

Experimental research into user reception and interpretation of symbols in websites would complement this study and help provide more definite conclusions about whether these emblems are being interpreted as localisers intended.

References


'Largest U.S. Corporations' (2005), Fortune, 151(8), F1 – F20.


Julie McDonough teaches in the School of Translation and Interpretation at the University of Ottawa, where she is currently pursing a PhD in Translation and Canadian Studies. A freelance translator and revisor, Julie has also participated in the University of Ottawa's Bilingual Canadian Dictionary Project as a French-English lexicographer. She holds an M.A. and a B.A. in Translation from the University of Ottawa and York University, respectively. Julie can be reached at julielaura.mcdonough@uottawa.ca.

Julie McDonough teaches in the School of Translation and Interpretation at the University of Ottawa, where she is currently pursing a PhD in Translation and Canadian Studies. A freelance translator and revisor, Julie has also participated in the University of Ottawa's Bilingual Canadian Dictionary Project as a French-English lexicographer. She holds an M.A. and a B.A. in Translation from the University of Ottawa and York University, respectively. Julie can be reached at julielaura.mcdonough@uottawa.ca.
Localisation in The Netherlands: Training and Career Opportunities

By Marcel Thelen and Han van de Staaij, Department of Translation and Interpreting, Maastricht School of International Communication, Zuyd University Maastricht, The Netherlands and Anne Klarenbeek, Dutch Translator, SDL Hengelo, The Netherlands

Abstract
This article gives an overview of localisation in The Netherlands, both in education and industry. The discussion on education is further narrowed down to the area of training institutes that offer courses on the translation aspects of software localisation; the discussion on industry comprises the whole spectrum. On the education side, the article gives an overview of localisation courses offered in The Netherlands and the tools used in such courses. On the industry side the article gives an overview of the localisation market in The Netherlands, i.e. its players, the systems used and produced, etc. The discussion also focuses on the participation of industry in training. Finally, one of the authors, who is a graduate himself, briefly discusses his expectations and experiences.

Keywords
localisation, translation, The Netherlands, education, training, industry

1. Activities Involved in Localisation
According to Esselink (2000:3) the following activities are involved in a localisation project: "(1) project management, (2) translation and engineering of software, (3) translation, engineering, and testing of online help or web content, (4) translation and desktop publishing (DTP) of documentation, (5) translation and assembling of multimedia or computer-based training components, and (6) functionality testing of localised software or web applications." These activities show that there are two distinct major roles to be played: engineering and translating. Each of these roles has a number of sub-roles. What these are can be derived from the more useful survey of the various aspects of localisation in Esselink (1998:6) where the people involved in a typical localisation project are given: "(1) Project Manager, (2) Translator, (3) Localisation Specialist/Senior Translator, (4), Proofreader/QA Specialist, (5) Localization Engineer, (6) Testing Engineer, and (7) Desktop Publisher."

For the education side of this article, the role of translating is understood to include – next to translating proper – project management and proofreading, and desktop publishing to a marginal extent only.

2. Education
2.1 Translation Training Institutes Offering Localisation Courses
The Netherlands has six major translation training institutes, one of which is the Department of Translation and Interpreting, Maastricht School of International Communication, Zuyd University in Maastricht. Of these six institutes, the Department of Translation and Interpreting in Zuyd University is the oldest (founded in 1981 by HM Queen Beatrix) and is the only one offering courses on localisation (i.e. both translation with the help of CAT tools and localisation in the strict sense). The objective of these courses is not only to give students an introduction to the various aspects of localisation, but also to serve as the basis for actual work using a wide range of localisation and translation tools; in particular the courses strive to train the students to become skilled users of localisation tools and novice translation professionals (see [2.2]).

2.2 Courses Offered
The Department offers a four-year course in translation and interpreting at BA level. In Year Two of this course, there are four modules on translation that serve to introduce and instruct students in the use of CAT (Computer Assisted Translation) tools, namely Trados/SDLX. Almost all translation work to be done after the introductory modules has to be done with the help of these CAT tools. At the end of the second year, students work as junior translators and revisors (for three-and-a-half weeks full-time) for an in-house simulated translation bureau that is staffed and run by fourth year students under the supervision of a senior lecturer. During this period of working for the in-house
translation bureau the students benefit greatly from using CAT tools (the use of CAT tools is made compulsory so that the students' work in the simulated translation bureau mirrors as closely as possible that of a real-life translation bureau). As for the regular exercises and assignments throughout the second year, CAT tools are becoming household tools more and more. The same holds for the third and fourth years of the course.

In Year Three, there is an introductory module on localisation proper, i.e. an introduction to localisation tools and working with these tools. Although culture and institutions are already part of the regular language programmes, culture also plays an important part in this introductory module on localisation – focusing on the aspects of culture that are present in localisation and in particular the technical side of these aspects.

The bulk of the work on localisation takes place in Year Four, where there is a further specialisation in localisation, which takes the form of a project. This project can be practical (i.e. on the actual localisation of help files, software and documentation/manuals) or more theoretical (i.e. on the comparison of various localisation tools, the evaluation of a particular tool, etc.).

As can be seen, by the time of their graduation all students are skilled users of CAT tools and have enough knowledge to work with localisation tools, with a number of them even specialising further in localisation. What counts is that both groups are prepared and ready to work in translation bureaus (or start one themselves) and have the skills needed to work with CAT tools and localisation tools.

2.3 Tools Used
During the second, third and fourth years of the course, a number of tools are taught and used. In Year Two, the CAT tools that are used are WordSmith, Trados MultiTerm, Trados Translator's Workbench (including the translation memory) and MultiTerm, TagEditor and WinAlign. There is also a course on HTML, albeit a basic course which reflects the ease with which students grasp this markup language. PASSOLO is covered in the introductory module on localisation in this year. The module contains a number of exercises on the use of this tool.

No further tools are introduced in the fourth year, but this may well change in the near future.

2.4 Input From Industry
Input from industry takes a number of forms. Currently these are:

1. Guest lectures on CAT tools, localisation and the industry,
2. Third-year work placement – for 19 working weeks – in a country where the first foreign language of the student is the language of habitual use, and
3. Fourth-year work placement – for 10 working weeks – most often at a professional translation bureau in The Netherlands, although it is also possible to go abroad. Both the third-year and fourth-year work placements are compulsory.

This input from industry will be expanded greatly in the very near future, after which it will also include the following:

4. Collaboration in the development of teaching materials,
5. Participation in the more commonly termed skills laboratory (the in-house simulated translation bureau), participation in the more commonly termed 'learning company', and
6. Placements for lecturers.

Participation in the in-house simulated translation bureau entails sending translation and localisation jobs for further processing and giving feedback on the products delivered.

The 'learning company' is a new phenomenon where the Department of Translating and Interpreting actively searches the market for (innovating, if possible) real-life projects of varying durations for students to work on. On successful completion of such projects, students will earn credits. The idea behind this is knowledge circulation: industry gains from the work that is done by the students for the Department and the Department will be able to enhance its knowledge by closely cooperating with the industry experts.

2.5 Cooperation With Industry
The Department of Translation and Interpreting actively seeks to cooperate closely with industry. The type of training given at the Department is vocational by nature. Therefore, it is one of the main objectives of the Department to cooperate with industry in the areas of the curriculum and placements – both for students and lecturers. As already described in Section 2.4, the Department is already rather successful in this respect (with plans in place to further expand industry input in the near future). See also Section 2.6 for further discussion on this cooperation. The Department already liaises with both localisation producers and localisation translation companies in The Netherlands and abroad. One of the objectives for the future is to give industry a greater role in the area of assessment.

2.6 Employability of Graduates
The Department of Translation and Interpreting has a number of instruments to measure the employability of its graduates.

Firstly, the third-year and fourth-year placements are very important factors in the employability of students. It happens very often that students doing their third-year placement are offered a job that will commence after their graduation, especially when doing their placement at a translation bureau. As for the fourth-year placements, it is a regular occurrence that placements lead directly to employment, with many placements being continued in the form of regular jobs. The Department works very closely with a number of renowned companies that offer such placements: SDL, Microsoft Ireland, Trados, various
'ordinary' translation bureaus in The Netherlands and abroad (mainly the UK), Lionbridge, Eclipse, RWS, and Philips Eindhoven. Also Medtronic (the world leader in medical technology) offers jobs to graduates at its translation and communication division in The Netherlands.

A second important instrument is the Department alumni scheme, central to which is an alumni website. More and more companies submit their vacancies for publication on this website and more and more graduates find jobs through this very same website.

It is noteworthy that one of the former graduates from the Department of Translation and Interpreting, Maastricht School of International Communication, Zuyd University in Maastricht has now become one of the world's leading localisation authors, namely Bert Esselink.

3. The Localisation Industry

The localisation industry has been growing rapidly and continuously in The Netherlands since the 1980s when the world witnessed the first personal computer, for which various types of content needed translation. The localisation industry received a boost a few years later when the first translation memories appeared, making translation much cheaper, faster and more consistent. The third boost came from the emergence of the Internet. Suddenly, data was accessible anytime, to anyone, anywhere. This opened up the international market for literally everyone, creating a huge growth on the translation demand side. And the market is still growing. It is a market that is growing for every area of the industry; and one that is growing constantly for all areas (notwithstanding seasonal peaks, e.g. higher sale of electronic goods at Christmas).

Over the past two years, various factors have contributed further to this growth. On the IT and multimedia side, we have the upcoming Microsoft Office 2007 suite and Microsoft Windows Vista operating system, plus the rise in sales of home networking products, gaming products and domestic appliances. On the automotive/mechanical engineering side, new EU environmental directives have led to the development of new engines and vehicles, and more localisation work as a consequence. Another factor is that companies realise more and more that they will lose out on sales if they do not continue or start localising their products.

And let us not forget the joining of the most recent EU member states, which has led to an even greater demand for localisation, on top of the growing list of European directives which necessitate the localisation of all sorts of content. Lastly, within some agencies, the Dutch language has been added to the so-called FIGS list (French, Italian, German, Spanish), forming the tier 1 of languages for all localisation work that has priority for most clients of localisation companies. Officially, however, Dutch is still a B-language though it is coming closer to the FIGS list. This move augurs very positively for the localisation industry in The Netherlands as it indicates that the demand for localisation into Dutch is growing.

3.1 The Market Players

Since the acquisition of Trados by SDL and the acquisition of Bowne Global Solutions by Lionbridge (both in mid-2005), SDL and Lionbridge really are the two main localisation players in The Netherlands. These two market leaders make use of freelance translators and translation agencies of all sizes for their outsourcing needs. Many enterprises in various industries also run their own in-house translation departments, but regularly call on freelance translators and translation agencies when their internal resources are fully booked. There is a great shortage of translators in The Netherlands – in particular in the localisation industry, thereby putting pressure on everyone at the supply end of the global information management chain. This shortage may be due to the growing demand for translation into Dutch (see the decision some agencies made to put Dutch on the FIGS list). The problem is that in a total population of 22 million Dutch-speaking citizens (Flemish included), there are not enough qualified translators. The shortage is also felt outside The Netherlands, e.g. at Microsoft in Dublin, Ireland where there is also a great need for native Dutch-speaking employees.

3.2 Expectations and Experiences of a Graduate

In this section, one of the authors, Anne Klarenbeek – who is a graduate himself – discusses briefly the expectations he had when he graduated and his experiences since then.

"Having graduated only four weeks earlier, I started working as an English-to-Dutch translator in August 2003. I quickly discovered that the pace was a lot higher than what we were used to at university. As I am working in a team that is specialised in the localisation of IT and multimedia content, I also noticed that, even though I had a greater than average knowledge of computer and networking hardware and software, I had a lot to catch up with. Personally, I found a great challenge (but also enjoyment and fulfilment) in jobs which require translators who are more skilled in 'transcreation' than translation – typically required for marketing pieces – and I noticed the same applies to newcomers who have joined the localisation industry over the past three years (albeit not everyone likes marketing pieces as much as others do). This area wasn't covered at university so I had to revert to my talents and the assistance of my co-workers.

My daily tasks also include file handling and resourcing. You could call it account management to some extent. My translation/review to account management ratio is around 70%-30%. This makes for a nicely varied pattern and a welcome change after a number of hours of concentrating on a piece of Help material or a user guide. The daily life of a localiser takes a lot of concentration and discipline and is often dynamic in the sense that one moment you are playing with words trying to sell a body groomer, and two hours later you are fixing the length of a handful of software strings, having just spent half an hour in-between
outsourcing work, issuing purchase orders and answering translators’ questions on the work they are helping you out with. There is never a dull moment if you like this kind of work.”

4. The Future

The prospects for localisation look promising in The Netherlands. Gradually, more training institutions are including localisation as a subject in their curricula, and in particular the Department of Translation and Interpreting of the Maastricht School of International Communication goes even further in that it is adjusting its curriculum to make it possible for industry to actually take part in the training of prospective localisers (see Section 2.4). In addition to this, The Netherlands can boast to have the world leader in localisation, namely Lionbridge, and the world number two, SDL. Lionbridge once started as a small Amsterdam-based localisation company named INK that gradually developed and expanded, changing its name once in a while until 1996 when the company became Lionbridge. Now the corporate headquarters are in Waltham, Massachusetts in the USA. The Amsterdam office is now a Lionbridge subsidiary. SDL is originally a UK-based localisation company, with its headquarters in Maidenhead. Over the past years SDL expanded and took over other companies, among them Alpnet in 2001. Since then SDL has a subsidiary in The Netherlands (Hengelo). Both Lionbridge and SDL attract the world’s greatest companies for localisation work: Lionbridge has the job of localising Microsoft’s Vista and SDL has the job of localising Microsoft’s Office 2007. Both companies are determined to strengthen their world position. All of these factors give the localisation industry an even stronger position in The Netherlands.

References


Marcel Thelen has been a senior lecturer in translation and interpreting at the Department of Translation and Interpreting of the Maastricht School of International Communication of Hogeschool Zuyd for the last 21 years. He is the director of the in-house simulated translation bureau and responsible for external relations. In addition, he is a placement supervisor. He can be reached at m.m.g.j.thelen@hszuyd.nl.

Han van de Staaij has been a member of staff of the Department of Translation and Interpreting of the Maastricht School of International Communication of Hogeschool Zuyd since its foundation in 1981. In 1981 he developed together with a colleague a terminological databank for educational purposes, which was later taken over by Elsevier. He introduced tools such as Trados, PASSOLO and WordSmith in the curriculum. He can be reached at j.m.vandestaay@hszuyd.nl.

Anne Klarenbeek has been a Dutch Translator at SDL International in Hengelo, The Netherlands since August 2003. He has a degree in English and French translation into Dutch from the Maastricht School of International Communication of Zuyd University, where he specialised in localisation and where he was the manager of the in-house simulated translation bureau, which formed his final project. He can be reached at aklarenbeek@sdl.com.
South Africa has a long and complex history. It has seen some of the oldest hominids known to modern man, tribes migrating from the north, colonialism by the Netherlands and the United Kingdom, imported slaves, several wars, and the political turmoil of the 20th century.

Currently South Africa has eleven official languages. It is second only to India in terms of the number of official languages. However, the South African language diversity is quite small compared to other African countries. Linguistically, the ten non-English languages are categorised as follows: one West Germanic language (Afrikaans) and nine languages from the Bantu family. Of these nine, four belong to the Nguni language group, three to the Sotho language group, and two are separate languages (Tsonga and Venda). Several of these eleven languages are also spoken in neighbouring countries.

In some parts of the country English serves as a 
lingua franca — mainly in the cities and more so to the south-eastern part of the country. Afrikaans is more dominant in the western part of the country and in rural areas. The use of the other languages is, to varying extents, localised in certain areas.

Translate.org.za was started in 2001 by Dwayne Bailey to localise Free Software for South Africa. A few applications were available in Afrikaans but no software was available in the other local languages at that stage. Having eleven official languages were mostly of symbolic value when it came to computers and technology.

1. Input and Display

There are relatively few technical problems with input and display for the eleven official languages. All use the Latin character set, with four languages using diacritics. The diacritics for three of these exist in some European languages. However, five characters exist that are unique to Venda. All of the extended characters had already been codified in Unicode.

No keyboard for South African languages has ever been developed.

To this day, many resort to 'Alt-codes' in Windows to input the extended characters, or use application specific character insertion techniques, or simply do not use the correct characters any more. Neglecting the diacritics is truly problematic, as this greatly reduces the morphological wealth of the affected languages. In some cases people still manually insert diacritics after printing, thereby making it impossible to have perfectly correct electronic copies. The situation has probably been worsened by the fact that the non-English languages do not enjoy high status in business and the fact that the diacritics do not occur in all of the languages, and in some cases do not occur with great frequency either.

To rectify this, Translate.org.za developed a keyboard with which all languages of South Africa can be typed [1]. It could not be much different from the standard US layout, since that is what is ubiquitous in South Africa, and many people need to use English regularly. It also had to be taken into account that many people would not use the extended characters often, or might not be adept typists.

Figure 1: A sample of the keys for the right hand on the South African keyboard layouts. The dotted circles indicate dead keys for using the relevant diacritics.

Along with the development of the South African keyboard, the popular DejaVu fonts were extended with the Venda characters that were still missing.

2. Initial Steps

As first projects, several smaller programmes were translated into the major languages of South Africa. Google South Africa was made available in four languages (unfortunately it is still not possible to limit searches to a specific South African language). The popular desktop environments, KDE and GNOME, were
translated and valuable lessons were learnt: skilled translators were not always easy to come by and managing translation efforts for several languages proved to be a mammoth task. Many translators preferred to work with spreadsheets rather than with applications supporting the standard Portable Object (PO) files that are used to localise Free Software. Experience has shown that translators need more training than might be expected.

Work started on the development of some tools as part of the Translate Toolkit to do format conversion and to test the quality of translations automatically. Some of the automated checks included tests for consistent punctuation, spacing, variable use and XML tags. These tools proved to be invaluable for the review process, as many messages could be filtered out for review by a non-native speaker of the language.

3. Locales

Locales were added to the GNU C library and to OpenOffice.org to support all official languages. All locales are now available in the Common Locale Data Repository. Microsoft has supported Afrikaans and South African English for some time and they added three more South African locales in Windows XP SP2. Locales for most of the other languages have been assigned, but will only be part of future releases of Windows. Locales for two of the languages are still lacking [2] [3].

4. Big Successes

In 2004 Translate.org.za released the first complete, localised Office productivity suite localised for South Africa by releasing OpenOffice.org 1.1 in four official languages [4]. As part of a large sponsorship Translate.org.za was able to extend the effort to update translations for OpenOffice.org 2.0 and include translations for all official languages [5]. The popular web browser and e-mail client from the Mozilla Foundation, Firefox and Thunderbird, were also translated into all official languages.

An important part of a fully localised office suite is a spell checker. Infrastructure was developed for the development of several spell checking systems and existing word lists were used to provide initial spell checkers. The complex morphology of the languages, especially for the languages in the Nguni group, offers severe challenges, especially for traditional UNIX spell checkers. The languages in the Nguni group are agglutinating languages, meaning that up to a whole sentence can be represented as a word. It is hoped that a future project would make it possible to extend Hunspell, the new checker used by OpenOffice.org, to provide support for the rich morphology of all official languages. Another difficulty with spell checkers is that they require good word lists. Often the best lists are to be obtained from lexicographers in the various languages. However, since the spell checkers are released as Free Software, it is very difficult to convince lexicography units – even though they are government supported – that this would not hamper their other commercial efforts in printed dictionaries.

5. Community Building

Part of the effort of Translate.org.za was not only to provide localised software, but also to ignite the flame of community localisation projects and to build a culture of multilingualism and of using localised software. Because English is also an official language, and English proficiency is relatively high amongst economically and technologically privileged, resistance to change has slowed the uptake of localised software. For some languages the community mailing lists are mostly dormant. The lack of translated teaching material has also been cited as an inhibitor for the adoption of localised software in training programmes.

To spark interest in software localisation, some Translate@thons (localisation sprints) were held with focus on specific languages. These events try to attract people to translate some software in a single day. The web-based translation tool, Pootle, created by Translate.org.za, has proven invaluable for these events. Such events can attract a mixed crowd in terms of translation skill, technical skill, and true interest. While a small group usually creates quality localisations, large numbers make it very hard to achieve quality translations and probably serve better for creating awareness and interest. Most people still consider computers and electronics to be something inherently English.

However, interest in localised software is slowly on the increase. The major accomplishments are reported on in local internet news sites, and some interviews were held on national and community radio stations [6] [7] [8].

6. The Afrikaans Localisation Community

Afrikaans arguably sports the most successful localisation community among the endemic African languages. It has active mailing lists, coordinated terminology efforts and many localisation projects undertaken by community volunteers.

It was possible to compile a reasonably good spell checker from previously compiled word lists and this was improved by community members. More recently this work was also extended with hyphenation rules and data for the AutoCorrect feature of OpenOffice.org.

The success of the Afrikaans community has indicated that effort is often required by an individual to take initiative, or to coordinate willing helpers. Without leadership many projects are unlikely to be completed or to achieve good quality. Without an existing volunteer community, newcomers find it hard to become involved. An existing community provides means for newcomers to join, without them needing to provide the initial leadership.

7. The Future

Although significant milestones have been reached, much remains to be done. The work on good spell checking for all languages was mentioned as an outstanding project; the complex morphology also affects development of an effective AutoCorrect functionality. Ideally grammar checking and thesauri should become feasible in the future. Collation specific for some of the languages should be considered, although the decision is a complex one, as powerful morphological analysis might be needed and could render such collating impractical.

Perhaps the most noticeable effect of the localisation work is to see how others have joined in. A few cellular phones are available in a few local languages, recently even with predictive text input for Afrikaans. A Zulu language interface pack for Windows XP was released in April 2006 and others would have followed soon thereafter [9]. Some of the Microsoft website is now also partially available in some South African languages.
Translate.org.za continues its work in the development of tools to simplify Open Source Software localisation; currently as part of the WordForge project. It is believed that this should afford even the smallest of marginalised languages a chance to efficiently manage their localisation projects at little or no monetary cost.

8. Conclusion

Despite common opinion to the contrary, we have proven that it is possible to localise software into all eleven official languages of South Africa. Only localising content for one of each of the language families is a common practice (a total of six languages), but localising for all eleven languages truly puts them all on equal footing. We have also aimed to translate complete user interfaces, rather than only translating the most commonly used messages.

We have shown that Free and Open Source Software can act as a driver for localisation. Almost nobody had anything on the cards a few years ago, but since Translate.org.za has delivered, proprietary vendors have at least started doing lip service and stopped denying that there is a demand for localised products.

On the downside, we have found that dominance of English in the economic sphere makes it hard for localised software to be adopted, as those who would use it, do not want it to hamper their career prospects. Furthermore, the dominance of proprietary software suppresses localisation. The use of proprietary software in government, education, the work place, etc., means that people don’t necessarily even have the choice to use software in their mother tongue, even now that it exists.

We expect Free and Open Source Software to continue to dominate in localisation. As proprietary vendors follow this lead and create even more awareness, we expect more people to be drawn to community localisation efforts, where much greater depth and breadth is possible. Harnessing the power of communities empowers both the languages and the communities, and builds a culture of multilingualism.

References


Friedel Wolff obtained his Masters degree at the University of Johannesburg and currently works for Translate.org.za as part of the WordForge project. He is also involved in the Afrikaans localisation community with spell checkers and grammar checkers as specific fields of interest. He may be reached at friedel@translate.org.za.