Using Translation Technology at Sun Microsystems

by Tim Foster
Software Globalization, Sun Microsystems, Inc.

Introduction

This paper explains how Sun Microsystems staff use translation technology for their translation activities on GNOME and other projects. It describes our experiences using open standards, such as XML Localization Interchange File Format (XLIFF) and Translation Memory eXchange format (TMX) and our experiences using tools to process these formats. In general, we found that these technologies can increase translator productivity and aid in sharing translations across multiple projects.

We will also discuss our translation editor, which has been developed in-house and has been in use for several months on real-world translations. In keeping with our tradition of supporting open standards, our editor can load and save XLIFF files and can export TMX files for use with other translation tools. We have received extensive feedback from professional translators who have been using the system in production, and we will explain some of the features that were added to accommodate their needs.

Finally, we will present our vision of translation technology and the advantages it can bring to increase translator productivity and translation accuracy.

Background

Like most large computer systems companies today, Sun translates its products into several languages in order to sell to a global market. In this paper we will share some of our experiences with large-scale translation projects.

In particular, we've seen that people working on the GNOME Translation Project (GTP) want to improve their translation consistency, tools, and processes, and we believe that our experiences with large-scale translation projects may be helpful. One technology area that we see as vital to increasing productivity for translators and increasing translation quality is that of standards. Using tools to process documents that conform to these standards has already proven useful at Sun.

The annual translation throughput at Sun is 40 million words (that is, we produce 40 million translated words per year). By GNOME standards, this probably doesn't sound like a huge amount, but it does take a significant amount of resources to meet this number.

Along with the translation cost comes the amount of time it takes to produce these translations. A typical translator will process 2000 words per day. Based on the number of words we translate, this amounts to 200,000 words per year. Clearly, with this annual volume of translations, we need a very efficient process to produce translated products as soon as possible after the English product ships. As we refine this process, we would like to ship localized products at the same time as the English product.

Along with the time it takes to do these translations comes a related problem: consistency. As mentioned above, the high volume of text to be translated is obviously too much for one translator (even one translator per language would find this volume an insurmountable hurdle). Ideally, we would have a team of translators working on a different section of the product at any one time. However, they must make sure to use the same terminology and translation style as their peers working on other sections of the product; otherwise, the result will be an inconsistent user experience.

So, to release our products on time, with high quality, and within a limited budget, we had to invest in translation technologies that would allow us to automate and streamline the translation process, to maintain or improve translation quality, and to improve workflow efficiency.

Translation Standards: Their Need and Availability

Translators today can expect to receive files in a number of formats, including but not limited to:

- HTML
- SGML
- XML
- Plain text
- PO files

Translators must be proficient with a wide variety of editing tools and file formats. If they don't have the software to read DocBook XML, for example, they have to obtain it and learn how to use it. Often, gaining the expertise to work with a new file format takes time that could be better spent actually translating. Moreover, people who write tools to process and manipulate the resulting files (for example, a tool that calculates word count that shows how much material has been translated and displays those statistics on a web page) are required to understand each file format. In other words, when a new file format is presented for translation, work is required by both the translator and the project co-ordinator or tools team to ensure that the new file format adequately fits into the existing project infrastructure--clearly an undesirable situation. One possible solution is to restrict file formats to only one format or a small number of formats. However, based on our experiences, restricting different product groups to a single file format is an uphill battle.

Jim Waldo, Distinguished Engineer at Sun, had some interesting thoughts about standards in a recent blog post titled "Why Standards?" (see http://www.artima.com/weblogs/viewpost.jsp?thread=4840):

> They [standards] started out solving problems. Because [standards] solved the problems, people used them. The use drove the standard, not the other way around. This allows innovation, this allows technical progress. Things that work get used by people who are trying to solve problems.

The problem of processing a wide range of file formats is an ideal case for a standard. Translators must be able to concentrate on their translation tasks, and tools developers need to be able to write a tool once without having to update it each time a new file format is introduced.

By employing translation standards in our tools, we solve both of these problems. Each time a new format is introduced, a filter is written to convert the format to XLIFF. After that, all tools that process translations can work with XLIFF documents and do not have to be changed each time another format is introduced. In addition, it enables our translators to choose the translation tool that they're most comfortable using. In our case, we created a translation editor that can read and write files in these translation standards, but there's nothing to stop translators from using another translation tool that supports the standard, or even write their own `emacs` or `vim` mode if they want to.

The two main standards we currently use at Sun in translation technology are XLIFF and TMX. XLIFF is a relatively new standard, designed specifically for the task of translation by a consortium of translation vendor companies and computer companies, and is now managed under OASIS (Organization for the Advancement of Structured Information Standards). An XLIFF file contains the translatable text from any input document. Once the XLIFF file is completely translated and reviewed, it can be converted back to the original document format.

XLIFF supports a number of translation-related processes, such as providing information about the context of the string for translation and supporting review-cycles and notes from translator, to name a few. One of the most useful parts of the XLIFF file format is that it allows for text to be "pre-translated"; that is, suggested translations can be inserted for each text segment being translated. This means that translation memory and machine translation tools can suggest translations for the translator to consider.

The other main standard that we use is TMX (Translation Memory eXchange). TMX was designed to allow translation memories (databases containing source strings and their translations) to be portable across translation memory tool implementations.

There is a subtle relationship between XLIFF and TMX. An XLIFF file contains the state of a translation in-progress, whereas a TMX file contains the completed translation that can then be imported into a translation memory.

Sun's Approach to Translation Technology

At Sun Microsystems, we believe in using and promoting open standards, and our translation activities are no exception. Our vision of translation is to employ technologies that improve the efficiency of the translation process, resulting in faster and better quality translations. Ultimately, we would like to improve the process to the level of getting translations "out of the wall"; that is, in the same way you access your bank account using an ATM, we would like to translate software by simply plugging the source files into a translation system and receiving the results immediately. We are gradually working towards this goal. Today, with the exception of our translation editor, all of our tools are server-side, allowing us to deploy the system on very large machines and databases where we can maximize system performance and make the system easier to manage.

Our use of translation technology starts at the point of content creation. Technical writers working on the source language document use a translatability assessment tool. Just as a spell checker flags incorrect spellings in the source document, the translatability assessment tool flags sentences that might be difficult to translate according to certain language rules. Each rule contains a sample sentence that suggests a rewrite of the sentence.

The next major piece of technology we use is a portal-based workflow system. This manages the whole localization process: source-file delivery from the engineering teams, conversion to XLIFF and partial translation by other translation tools, and delivery of the partially translated XLIFF files to various translation vendor companies throughout the world. These vendors connect to the portal, select files, and start translating the files. Once the translation is completed, the XLIFF files are then uploaded onto the portal, which sends the files back to the engineering groups for inclusion in the final product.

We also use a glossary management tool. This stores source language terms in a database as well as the translations that have been approved for those terms. Using a glossary management tool allows translators working on different documents to look up the correct translation for each term, thereby providing a better experience for end-users and a more consistent translation.

We are investigating the use of machine translation tools in our translation workflow. The other major tools in this workflow process are the translation memory software and the translation editor. We'll cover these in more detail in the following sections.

Translation Memory System

Our translation memory system is a key component in our translation technology tool set. It consists of two main components. As mentioned earlier, the first component is a large database containing source segments and their translations in a number of languages. Each segment in the database is usually a sentence, but it could also be a paragraph or a phrase. The segments in the database are stored with some metadata, describing the context of the segment (for example, which source document it came from or which engineering group created it).

The second major component of a translation memory system is a fast fuzzy search mechanism. Using this, we can search the translation memory for the segments in an input document and produce a partially translated file. Every time we find an exact match in the database, we can store that translation as a suggestion in the output document. We can also look for fuzzy matches in the database. Fuzzy matches are similar to the input segment, but don't match exactly. We can add fuzzy matches to the XLIFF file and include data about the quality of the match for the translator to use when completing the translation.

When searching against the translation memory, we can narrow the search using the metadata stored in the database. This limits the search to documents that are in the same category as the input document or that have been produced by the same product group. Of course, there's a trade-off between the number of matches we could achieve versus the usefulness of those matches to the translator; returning several matches per segment may actually slow the translator down, as she will need to search through those candidate translations for the correct one.

Again, localization standards are used by this tool when possible. The input and output documents for the translation memory are in XLIFF format: the input contains the source language segments and the output contains the source language segments and suggested translations for those segments in a single target language. Also, the translation memory system can import TMX documents, adding to the number of segments in the database.

It is useful to be able to keep track of formatting information in the original document. XLIFF allows us to mark formatting sections in the original document and preserve them throughout the translation process. This feature is very useful because we can easily take account of formatting when computing how "fuzzy" a match is. For example, the segment:

> This is a <emphasis>small</emphasis> sentence.

is quite similar to the segment :

> This is a small sentence.

Although the two segments appear to be about 60% similar, if we disregard the formatting in those segments, they are exactly the same. To maximize cross-format leveraging (for example, obtaining matches from a DocBook translation when the source document was HTML), we employ special algorithms to weight matches so that the difference in formatting sections does not influence the final percentage-match figure as much as differences in the content of the strings. These segments would still be considered different, but according to our match algorithm, they turn out to be 96% similar.

As the translation memory system is the first tool used in our translation workflow, for convenience we have also built in a set of filters to convert documents in HTML, DocBook, plain text, and XML to XLIFF. We are also working on a number of software-message file filters.

Translation Editor

The translation editor is the main interface that translators use to translate XLIFF files.

The translation editor shown in Figure 1 is split into three main sections. Segments in the source language document appear in the left column. Segments in the target language document appear in the right column (this layout can be changed). The suggested matches for the current segment appear at the bottom of the screen, along with any metadata about the match and an indication of the quality of the match found. The suggestion area highlights the difference between the match found in the database and the actual segment for translation. Clicking the Transfer button moves the selected suggestion into the target language area, where the translator can complete the translation.

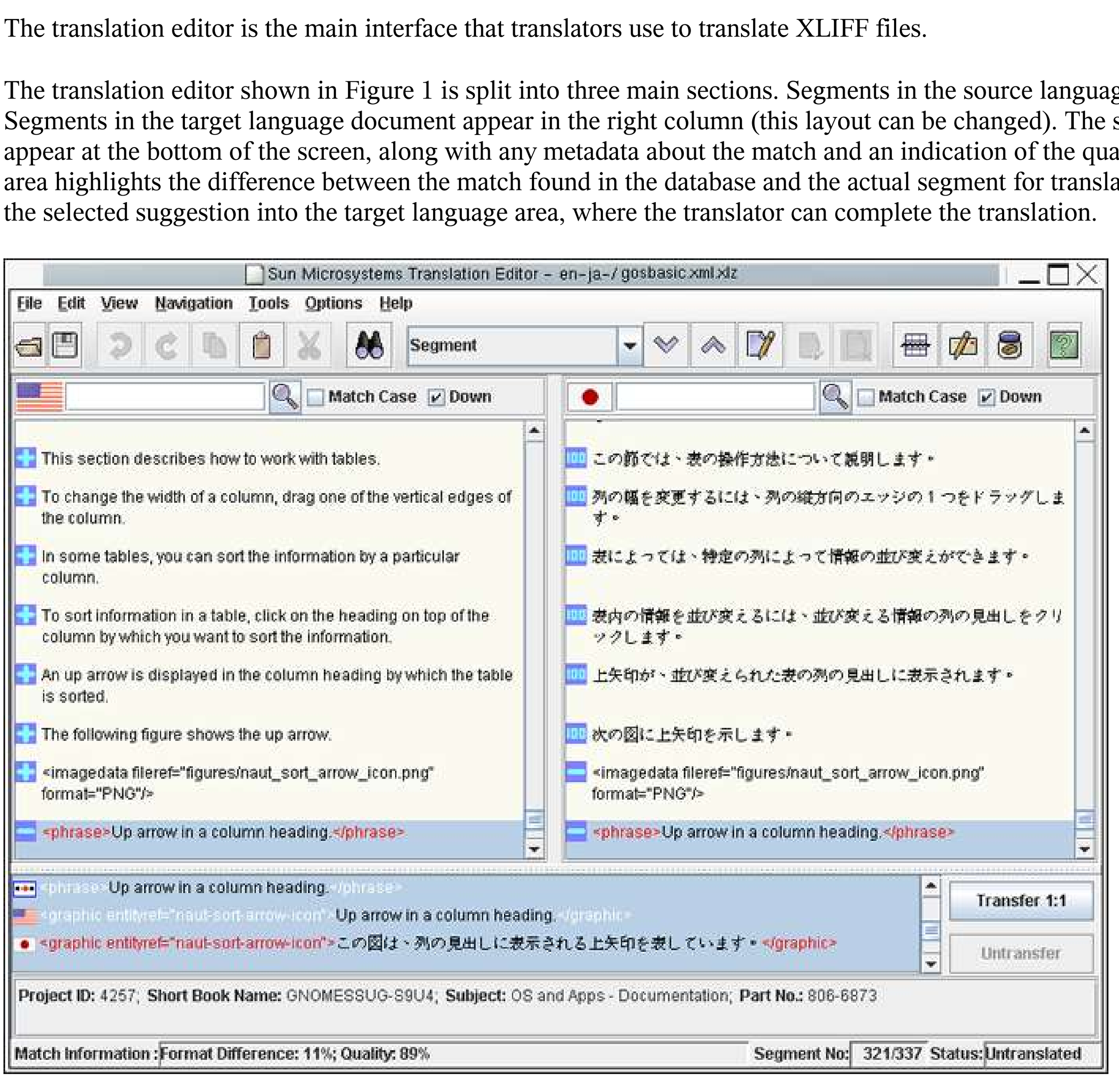

Figure 1: The Sun Microsystems Translation Editor

While developing the editor, we sought feedback from our translation vendors and feel confident that it fits the needs of these users. As with any end-user-focused application, the editor had to be simple and intuitive. It is easy to configure and provides all of the features that a translator would expect. During consultation with our users, the following features were requested frequently and have been implemented in the current version of the editor:

- Format checking--Formatting that is in the target segment can be checked against the formatting in the source segment to ensure the translator hasn't omitted or included any extra formatting (bold, italic, and so on).
- Integrated "mini TM"--When the translator encounters a segment that has been previously translated in the file, the segment will be translated automatically. Also, any sentence that has been translated in the past that is similar to the one currently being translated will also be suggested as a match.
- Translation status marker--Translations can be marked as "complete," "for review," or "approved," to aid in the translation/review process.
- Translation comment marker--A translator can add comments about any segment. The comments can be useful in the translation/review process.
- TM match information--Each translation memory suggestion contains information about how close the match is to the source segment and also displays any metadata that was stored with the proposed translation.
- Printing--We can produce an HTML-formatted version of the XLIFF file at any time for printing and review.
- Easy navigation--Commonly used keyboard shortcuts can be modified to suit the translator's preference.
- Back-conversion--For ease of use, translators can choose to convert the XLIFF file back to the original file format so that they can check the layout/formatting of the finished translation.

The Future of Translation Technology at Sun

Despite the translation tools that are already in production at Sun, we feel that there is still work to be done to improve the translation process further. With XLIFF, we have an excellent way to add more translation tools into the translation process. We would like to integrate the tools we have so that translators can use a single interface to all of the translation tools.

As mentioned earlier, XLIFF allows us to provide translation suggestions from more than one tool. We envisage a workflow in which a translation memory tool would add the first set of translation suggestions to the XLIFF file. A next stage in the process would be to perform "term-mining" on each segment. This would identify possible terms for translation, which could be automatically looked up in the translated glossary, and these terms could be highlighted by the translation editor. A final stage of the translation workflow would be to pass each segment into a machine translation system and add those translations into the XLIFF file. All of this work would be completed before the file reaches a translator.

On the client side, we believe that a networked group of translators could be more efficient than a group of standalone translators. We want to investigate additional features for the translation editor that would create a peer-to-peer network of translators, so that completed translations are made available to other translators connected to the network, thus improving the efficiency of the group.

The standards we are using at the moment make this work easier, but other standards will be important as we add new tools into the translation process.

TBX is a standard file format for transferring glossary information between translation tools. OLIF (Open Lexicon Interchange Format) is a standard for transferring terminology between machine translation and other NLP tools. We believe that these standards will also play a part in our vision of a translation workflow.

A final effort worth mentioning is being worked on by the OASIS Translation Web Services TC, which will further enable tools from different groups to work together.

Conclusion

In this paper, we have shared our experiences with translation standards and have expressed our belief that the use of common standards in the translation space is something worth investing in. We hope that the many people working on free and open source software will find them of benefit to the different projects that they are working on. We have also demonstrated the use of a dedicated, standards-based translation memory system and shown some of the features in a standards-based translation editor that are useful to translators when translating software and documentation.

Resources

XLIFF: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xliff

TMX: http://www.lisa.org/tmx

OLIF: http://www.olif.net

TBX: http://www.lisa.org/tbx

OASIS WS TC: http://www.oasis-open.org/committees/trans-ws/charter.php

Article on XLIFF: http://developers.sun.com/dev/gadc/technicalpublications/articles/xliff.html