Experiences on an Open Source Translation Effort in Japan

Hiroki Sato Keitaro Sekine
Faculty of Science and Technology, Tokyo University of Science, JAPAN
FreeBSD Japanese Documentation Project
hrs@jp.FreeBSD.org

Abstract

As network connectivity becomes more world-wide, the importance of translation efforts among open source software projects has grown rapidly. Many projects, including FreeBSD, NetBSD, and OpenBSD, already have some teams responsible for translating documents into other languages, but there seems to be few reports in respect to problems, efficiency, and so forth around the projects.

Translation work has several common characteristics and problems that are obviously different from typical software development. This paper describes them through my experiences with FreeBSD Japanese Documentation Project (doc-jp) for last two years, and points out the tasks that are required for such work.

1 Introduction

In countries where English is not the mother tongue, many people mainly get information about open source software via translated documents and have development activities in their own language. This is because of the fact that a lot of open source projects use English as the common language. As open source projects extended, their translation work [1–3] is as important for people as their English documentation efforts are.

In many cases, the translation efforts are not enough, since they need skilled people who understand at least two languages and know the software itself and some problems specific to translation. Things such as security advisories, which must be translated and spread all over the world as quickly as possible can make translation work more difficult than one had expected and can result in undesirable situations.

For this reason, we should consider efficient methodology on such translation work as part of a software development project. Through this paper, I will discuss several common characteristics and problems of the translation work that are obviously different from software development through my experience.

2 FreeBSD Japanese Documentation Project

2.1 History and Overview

To begin with, I would like to introduce a translation project which I belong to, which is the FreeBSD Japanese Documentation Project [1] (doc-jp). The project was started by some FreeBSD developers in Japan in 1995, and it plays an active part in translation of FreeBSD-related documents from FreeBSD Documentation Project [4] (FDP) into Japanese and the management other Japanese documents. Another project, the FreeBSD Japanese Manual Project [5] (JPMAN) is responsible for translating FreeBSD manual pages into Japanese. JPMAN and doc-jp complement each other, and are two primary translation efforts of FreeBSD in Japan.

The results of doc-jp's translation work over five years are as follows:

- FreeBSD Handbook (1996-, maintenance phase)
- FreeBSD FAQ (1997-, maintenance phase)
- FreeBSD Tutorials (1997-, maintenance phase)
- www.FreeBSD.org (1997-, maintenance phase)
- FreeBSD Release Notes/Errata (1997-, per release basis)

- FreeBSD Security Advisory (2000-, per release basis)
- Online version of “The design and implementation of 4.4BSD Operating System, Chapter 2” (2001, maintenance phase)
- FreeBSD-announce (important announcements only)

Most of the above are now actively maintained. The classifications of the work phases are described later.

The basic work of doc-jp is quite simple. A translator fetches a document in English via CVS, and then translates and submits it. Other project members review the translation and discuss it as the need arises. These processes are mainly dealt with on the doc-jp mailing list. When finished, a FreeBSD committer commits the translation into the FreeBSD CVS repository.

3 Characteristics of the Work

As described above, the process of translation work is similar to typical software development in a sense, but it seems that there are remarkable differences among them. In this section, several characteristics specific to translation that I noticed through doc-jp’s work are described.

3.1 Review and Quality Evaluation

As you know, computer software runs on a computer, so the primary quality and functionality of code are determined objectively by their behavior themselves. Quality and functionality of translation, however, are obtained only by the review of project members and readers. Unfortunately, there are few software tools that can help us.

There are various kinds of translation “bugs” (e.g., typos, mistranslations, and so on), and to translate technical terms that are not familiar into Japanese requires “good taste” in translation efforts since it is subjective and sometimes leads to a conflict of members’ opinions, which in turn can make translation more difficult than that of a programming project.

In short, the work not only needs good translators, but also good reviewers. This problem can affect a small project more than a larger one because the number of bilingual people involved is most likely less than in a large project.

3.2 The Life Span of Translated Documents

Another problem that can arise is the fact that original documents can be revised. Naturally, old code is also improved upon, but we can still use the old one if no fatal bugs exist. Older translated documents, on the other hand, usually cannot be used since in most cases the original ones are revised when they are obsolete or no longer reflect reality.

The general belief is that he translation work is to translate existing English documents into another language. While this is certainly true, it is even more important to maintain the documents after translation. Translated documents that disagree with the originals are of a negative value only, so the translators must avoid such situation. And, as with any open source project, keeping project members’ motivated is extremely essential. The translation of documents is very hard work. Frankly speaking, maintenance of translated documents is not very creative for people that can understand the English document. After all, translated documents are no different than the original English documents.

3.3 Classification of Translation Phases

A translation work can be classified into the following three phases; first translation phase, maintenance phase, and feedback phase.

The first phase is to translate an English document into Japanese. This is called “the first translation phase.” Once translators are collected, their motivation is relatively high. Work usually goes well in this phase until a first draft is submitted by the translators.

The second phase, called the “maintenance phase,” is when submitted translations are reviewed and refined. This is the hardest and the longest job of the three.

The third phase is “feedback phase.” When the translated documents are revised enough and become stable, frustration to fix bugs in the original documents arise by degrees. It is more or less worth revising original documents that are difficult to understand for translators, so the work could be effective as “good review” for the original ones.

When this classification is applied, most of the translated documents that already exist are in the "maintenance phase." Again, it should be emphasized that translation work is not temporary work. "Review and refine" in the second phase have the most relative importance throughout the work, making translation work as much of a continuous effort as the software development.

3.4 Statistics

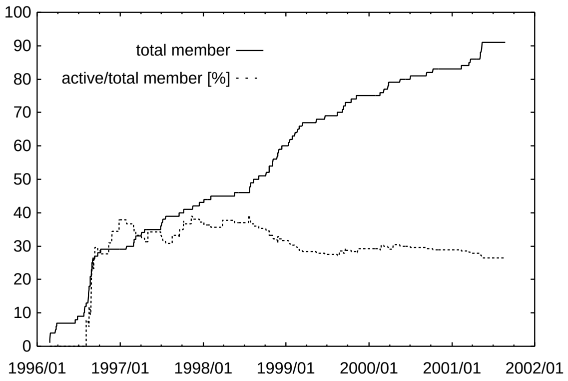

Figure 1: The growing number of doc-jp's members in proportion to active members.

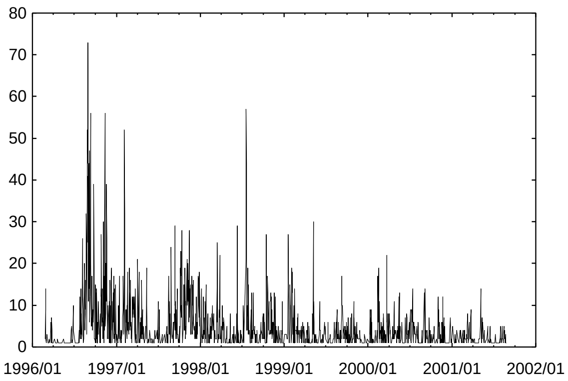

Figure 2: Distribution of the amount of mail on the doc-jp mailing list.

Figure 1 shows the growing number of doc-jp members in proportion to active members (defined here as a member who has posted more than 50 times to the doc-jp mailing list). Figure 2 shows the amount of email, on the doc-jp mailing list over a span of the last few years. Figure 3 shows the distribution of the translation phases for a particular document.

As shown in the figures, the member is growing, but the number of active persons still is few. A few of the members have continued to work regularly over a long period of time. Most of their interests concentrate on the first phase, yet the second phase has the most relative importance of the three in respect to the amount of work. Contrary to what you may think, the first phase takes a relatively short amount of time when compared to the other two.

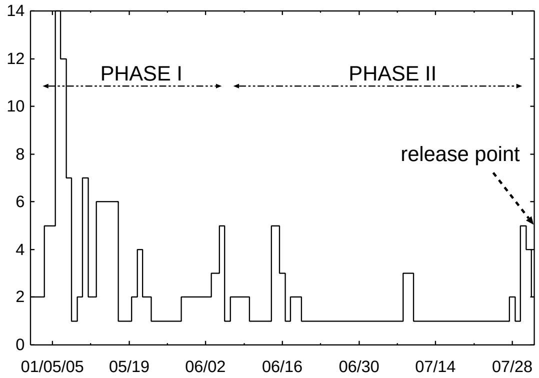

Figure 3: Distribution of the number of mail and translation phases for a certain document.

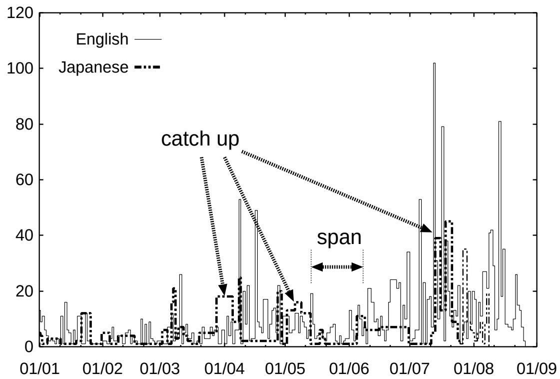

Figure 4: Commit frequency to FreeBSD doc/ and www/ tree.

Lastly, Figure 4 shows the commit frequency to the FreeBSD CVS repository's doc and www trees that FDP and doc-jp are responsible for. Notice that the FDP's work progresses regularly, but doc-jp's work is relatively intermittent. This is because the active members make the translated documents catch up with the originals when they are behind.

We have translated over 60% of the original documents in FreeBSD source tree into Japanese as of November 27, 2001, and have actively maintained them. A simple breakdown of them is shown in Table. 1.

Table 1: Percentages of translated documents.

location of the source tree	en	ja	%
doc/${LANG}/articles	26	8	30.8%
doc/${LANG}/books	72	47	65.3%
www/${LANG}	203	153	75.4%
src/release/doc/${LANG}	22	7	31.8%
Total	323	215	66.6%

4 Experiences and Problems

In this section, I will show several problems raised around the work. They may be relatively biased, but they are from real experiences.

4.1 Working Style and Release Engineering

As far as I know, some translation teams have their own CVS repositories. Doc-jp had its own repository in the past, but merged everything into the FreeBSD CVS repository three years ago because of the time lag before merging the results.

This doubled developing model undoubtedly has some advantages. Two such advantages are the main CVS repository is no damaged in the event of a mistake, and that non-committers can use the repository. Although I can't say that which model is preferable (working in the main repository or a separate one), both of them need a certain amount of time to merge their results into the main tree.

Roughly speaking, the release engineering process of FreeBSD is as follows. First the source tree is "frozen" and only the release engineers can make changes during the freeze period. This process usually lasts a few weeks to a month to work out any fatal bugs, and then it will be released. During this process, documentation is also prepared in parallel, however most release-related documents, such as release notes, etc., are prepared just before the release point, so we are usually pressed for time to translate them.

You might think that the translation should not be included with the release, however most translation teams would like to see the release documentation in their native language and have the documentation included. Under some circumstances, the translation cannot be prepared in time due to the lack of members and the lack of time available. As shown in Figure 3 and Figure 4, the translation work usually needs at least one or two weeks to catch up with the original documents.

4.2 Toolchain

Today, most of the documents in the FreeBSD source tree are marked up as DocBook/SGML or XML, so in order to make them readable, we need a toolchain to process them appropriately. Naturally, doc-jp uses the same toolchain used as the FDP, however this has raised some problems.

As you know, non-English languages have a specific encoding scheme. For example, Japanese has EUC-JP, ISO-2022-JP, and ShiftJIS (also known as MSKanji). All of these consist of 8-bit characters and many toolchains do not support such encodings, so we must find a way to make it work.

For example, Jade, the DSSSL engine which can output documents in several formats (HTML, PostScript, PDF, etc.) can produce Japanese output, but its $\TeX$-based backend which is used for Postscript and PDF output, does not work properly. I am working on this issue, however since it is not completely, a PostScript or PDF version of FreeBSD Handbook in Japanese is not available.

You may think that unicode is the solution. While this may be true, it only solves part of the problem. There are few tools that are currently available which support unicode. In addition to this, from a Japanese point of view, unicode is not sufficient for representing Kanji characters.

4.3 For English Writers

This section discusses the kind of sentences and CVS operations that translators have the most trouble dealing with during translation efforts.

Figuratively speaking

While good sentences often include a figure of speech, translators tends to run into trouble when dealing with such complicated expressions. Most translators always expect sentences to be simple, straightforward, and to the point without going about so in a roundabout way.

A typical example is using a joke. Do you know of any jokes that people across the world can understand? It is extremely difficult to translate jokes into non-English languages and retain the humor.

Another example is slang, which does not appear in a dictionary. They also make translation work very difficult. I never insist that jokes should be kept out of documents, however, they should be kept conservative for good understanding.

It is also preferable to write full sentences and not one or two word phrases. Some experienced translators can understand such expressions, but it tends to mislead the translators. Remember, simple is preferable.

Unfortunately, I cannot give many examples because the reader would need knowledge of both Japanese and English in order to understand them, but I ask that you remember this; if your message is valuable, it can be translated even if it is a post to a mailing list or newsgroup. In order to increase the chances to find folks in other countries interested in your message, use sentences which they can translate smoothly.

The Rule of "Separate Commits"

In CVS, original documents should follow the rule of "carefully separated commits." This means that any commit to original documents should be divided into cosmetic changes and content changes. If the two are not divided properly, the diff deltas generated by CVS grow unnecessarily large. Most cosmetic changes have nothing to do with translation, so translation teams always appreciate separate commits because it reduces the amount of the work needed placed upon them.

For example, consider there is a SGML document that consists of two paragraph enclosed with `<para>` as shown below.

```
<para>This is a sample document marked up
  as DocBook/SGML.  If you are familiar
  with HTML, understanding SGML documents
  is not difficult.</para>

<para>Now, consider what kinds of
  difficulties there are in management
  of SGML documents.</para>
```

When a commit is made that changes the spacing of the first paragraph, the document and delta generated by CVS could look like this:

The modified document:

```
<para>This is a sample document marked up
  as DocBook/SGML.
  If you are familiar with HTML,
  understanding SGML documents
  is not difficult.</para>

<para>Now, consider what kinds of
  difficulties there are in management
  of SGML documents.</para>
```

The delta:

```
 <para>This is a sample document marked up
-  as DocBook/SGML.  If you are familiar
-  with HTML, understanding SGML documents
+  as DocBook/SGML.
+  If you are familiar with HTML,
+  understanding SGML documents
   is not difficult.</para>
```

If the "separated commits" rule is not followed, the translator should carefully compare deltas like those seen above. This is a very difficult and wasteful effort. A simple sign such as "cosmetic changes only" in the CVS log and a carefully separated commit greatly helps us reduce the amount of the work in translation.

The rule of "separate commits" originated in the FDP for this reason. Recently, I have noticed another situation that gives translators trouble. The following is the same two sentences as in the above example, however, they are interchanged. This often occurs when documents are being re-organized.

```
<para>Now, consider what kinds of
  difficulties there are in management
  of SGML documents.</para>

<para>This is a sample document marked up
  as DocBook/SGML.  If you are familiar
  with HTML, understanding SGML documents
  is not difficult.</para>
```

This change generates the following delta:

```
+<para>Now, consider what kinds of
+  difficulties there are in management
+  of SGML documents.</para>
+
 <para>This is a sample document marked up
   as DocBook/SGML.  If you are familiar
   with HTML, understanding SGML documents
   is not difficult.</para>
-
-<para>Now, consider what kinds of
-  difficulties there are in management
-  of SGML documents.</para>
```

While this does not seem to be a widely known fact, the interchanging of sentences can confuse translators very much. It increases the size of the delta that the translators must examine, and can very difficult to understand. You can imagine such interchange occurs in more complicated way.

I suggest that the changes described above should be considered cosmetic changes and separately committed from content changes. CVS logs help very much in these situations, so when such a change is made, please take what sort of change it is into account when writing the CVS commit message.

In short, as translators, we hope that those writing documentation will pay more attention to the translation work. If this is done, the documents will be able to be read by a larger amount of people, which is also good for the project.

5 For Efficiency

Up to this point, I described problems involving the both the translation work and the parent project. Next, I will show several ways for efficient translation work itself.

5.1 Providing Text Fragments But Whole

It is difficult for all of the translation project members to determine the status of the original and translated documents, so sometimes they hesitate over which to choose. If split documents are provided positively for the project's mailing list and so on, they can translate and review them immediately without unnecessary trouble.

Original documents that will be translated should be divided into relatively small text fragments and provided to translation project members. In addition, it is better for reviewers to keep the translated and the reviewed document side by side with the original text so they can easily compare the two.

5.2 Infrastructure

In doc-jp, translators need to fetch a target document via CVS, but doing so is sometimes difficult if the translator has no experience with CVS. Thus, an interface supporting translators with target documents and translated documents to be reviewed should be prepared. For instance, there is an experimental one for doc-jp [6], and other projects have similar facilities [7–9]. In particular, [7] is more functional since it includes reservation of translation.

Finally, older documents already translated should be marked so that people who read them are aware of their status. As mentioned earlier, obsolete documents are nothing but harmful for everyone. In doc-jp, a revision checker [10] is used for build process of the translated documents.

The revision check mechanism realized by [10] is quite simple. Original documents surely have a line of CVS ID like this (actually this is one line):

```
$FreeBSD:
doc/en_US.ISO8859-1/books/handbook/book.sgml,v  1.119
2001/11/19 11:38:45 murray Exp $
```

And we make the translated documents have a line indicating its parent document as shown below:

```
Original revision: 1.119
$FreeBSD:
doc/ja_JP.eucJP/books/handbook/book.sgml,v  1.70
2001/10/27 18:12:06 hrs Exp $
```

The revision checker compares the CVS ID of the original document with the "Original revision" line in the translated document and the result is reflected in the definition of an entity called `%rev.diff;` as "`IGNORE`" or "`INCLUDE`" which used for a marked section of SGML. Actually, when the two revision is matched:

```
<!ENTITY % rev.diff 'IGNORE'>
...
<![ %rev.diff; [ this document is obsoleted! ]]>
```

and when they are not matched:

```
<!ENTITY % rev.diff 'INCLUDE'>
...
<![ %rev.diff; [ this document is obsoleted! ]]>
```

When the documents are rebuilt, this definition is included into each documents, so the documents themselves can notify the reader and maintainer that the translation is not up-to-date.

The important things are: 1) keep translated documents as up-to-date as possible, and 2) if circumstances do not allow this, notify the readers that the translated document is not up-to-date. The simple revision checker described above does just that.

5.3 Word list

During translation, we often think—especially when it is one of the technical terms—"what does this word mean?" To make things easy, we maintain a translation word list. Generally, it is a list which includes original and translated words on a word-by-word basis. Personally, I think there are problems with this and it is not sufficient.

First, maintenance of the list is relatively hard work. While many people translate documents, how do we determine which words should be candidates for the list? We have to discuss it, and the discussion usually takes quite a bit of time. Moreover, the objective answer is not always obtained.

Second, translation of sentences always goes with the sentence's context. The list of translation words does not include the context, it is possible to mislead the translators.

However, it is also undoubted that the words list is useful to keep consistency of translated words. The primary disadvantage is that it increases the project's work, and our goal is not to make a comprehensive word list.

I am designing an alternative that will identify already translated documents and includes a full-text search engine. Although it is not finished at the time of writing, using this method will allow translators to find the word they are looking for and the output will include a translated example sentence. Since the results of translation work are used as a database, I believe that the trouble described above can be somewhat relieved.

6 Summary and Future Directions

In this report, characteristics and problems specific to translation work are described through my experiences. To think little of translation efforts or to regard it as normal software development is the wrong idea.

I think that the translation efforts in various projects need much more technical cooperation and information exchange about their efficient management. The majority of frameworks can be shared, and the maintenance of them, such as word lists and style guides, can be done cooperatively instead of reinventing the wheel. The primary objective of the work is translation and not providing infrastructure itself.

With such a goal in mind, I have made a proposal for a project called the "Doc-ja Archive Project [11]," which supports various Japanese translation projects in early 2001. Unfortunately, the project virtually has obtained no results thus far, but we hope to become a place for discussion of translation efforts in Japan.

7 Acknowledgments

I thank Japan FreeBSD Users Group and FreeBSD Japanese Documentation Project for supporting my translation activities.

References

[1] FreeBSD Japanese Documentation Project, *"FreeBSD doc-jp web page,"* 1999
http://www.jp.FreeBSD.org/doc-jp/

[2] www.NetBSD.ORG Japanese Translation Project *"www.NetBSD.ORG Japanese Translation Project web page,"* 1999
http://www.jp.netbsd.org/ja/JP/Project/www-ja/

[3] www.OpenBSD.org Japanese Translation Project *"www.OpenBSD.org Japanese Translation Project web page,"* 2000
http://ja.open.4bsd.org/

[4] FreeBSD Documentation Project, *"FreeBSD Documentation Project Primer,"* 1999
http://www.FreeBSD.org/docs/en_US.ISO8859-1/fdp-primer/

[5] FreeBSD Japanese Manual Project, *"FreeBSD JPMAN web page,"* 1997
http://home.jp.freebsd.org/man-jp/

[6] FreeBSD Japanese Documentation Project, *"Synchronization status for FreeBSD Documentation Project,"* 2000
http://www.jp.FreeBSD.org/doc-jp/syncstat/

[7] FreeBSD Japanese Manual Project, *"JPMAN web reservation system,"* 1997
http://home.jp.freebsd.org/man-jp/yoyaku/

[8] Linux Japanese FAQ (JF) Project, *"JF in progress,"* 1999
http://www.linux.or.jp/JF/workshop/JF-in-Progress.html

[9] Debian Description Translation Server Project, *"Announcement of Debian Description Translation Server being available (in Japanese),"* 2001
http://lists.debian.or.jp/debian-doc/200108/msg00006.html

[10] FreeBSD Japanese Documentation Project, *"A script for SGML preprocessing and revision checking,"* 2000
http://cvsweb.FreeBSD.org/www/ja/prehtml

[11] Doc-ja Archive Project, *"Doc-ja Archive Project web page,"* 2001
http://openlab.ring.gr.jp/doc-ja/