TRANSCRIPTION SUPPORT SYSTEM USING SUBVERSION

Takehiko Murakawa, Hitoshi Fukuoka
Daichi Noda and Masaru Nakagawa
Faculty of Systems Engineering, Wakayama University, Japan

Keywords: Digital library, Ancient document, Web service, Version control.

Abstract: We report the data management system and the interface for reading the shot image and the text of a Buddhist sutra written in Chinese and modifying the text so that it may be the same as the image in terms of content. By using Subversion we maintain the text files efficiently and obtain the difference of the contents between any two points of time easily. To make sure that the system can be employed as a multiuser transcription support tool, we present the working model and deliver the experiment where the workers used the system and we found revision markings on which two workers or more made. Furthermore we propose the method for piecing together the workers' respective outcomes to produce the integrated text file.

1 INTRODUCTION

Tens of thousands of Buddhist sutras which were transcribed by hand around 1,000 years ago are existent in Japan now. Through the literature assessment, it became evident that those sutras reflect the earlier Buddhist canons in Chinese, in comparison to the set of sutras by woodblock printing after the 10th century. To prevent such monumental documents while the further research is permitted, a project was organized and has received the subsidy from the government of Japan to enable the researchers to computerize the materials and manage them with databases. The authors have worked on the project with a view to providing the database system together with serviceable interfaces (Tanaka et al., 2006; Fukuoka et al., 2009).

When bending our eyes on the activities outside Japan, the conservation of ancient documents including transcribed Buddhist sutras has a close connection to the cultural heritage. European countries seem to be more active than the United States and Asian nations. For example, the report of EPOCH (European Research Network of Excellence in Processing Open Cultural Heritage) (EPOCH, 2005) insists on the importance of the prevention of historical materials in various languages, although the report mainly describes the maintenance of building structures and remains. Various documentation and reading supporting systems were actually constructed (Chen et al., 2007; Ribeiro et al., 2007; Alshuhri, 2008).

There exist temples in Japan say Kongoji Temple and Nanatsudera Temple which possess lots of Buddhist sutras in good condition. The experts of digitization have had the approval of the head priests to use the rooms in the temples, brought photographing equipments including high-resolution digital cameras, photography platforms and lighting apparatus, and taken the pictures. Most sutras of Kongoji Temple have been taken, and the amount of the file size rose above 1 terabyte.

During the photo shoot, the lighting and the avoidance of distortion were sufficiently taken into consideration. In addition, we developed tools for combining the shot images of the sutra, as well as the Web application which allows one to browse the sutra images and move the viewpoint freely by the drag operation just as Google Maps. The trouble is that he or she cannot make use of full-text search on those sutras. The authors attempted to apply several character recognition applications to the images, but did not lead to a positive outcome. There is, fortunately, released a data set of the text on Buddhist sutra in Chinese. The CD-ROM image file of the text data is available free of charge, but we had to pay attention to the fact it was derived from the woodblock-printing sutras. When looking at the shot image and the corresponding text file for comparison, we detected plenty of mismatches. If the editing interface is supplied say by extending the sutra browser mentioned above, then the user will be able to find the desired sequence of Chinese characters using the full-text search. More-

Murakawa T., Fukuoka H., Noda D. and Nakagawa M.
TRANSCRIPTION SUPPORT SYSTEM USING SUBVERSION.
DOI: 10.5220/0002846801500155
In *Proceedings of the 6th International Conference on Web Information Systems and Technology (WEBIST 2010)*, page
ISBN: 978-989-674-025-2

over the information about the mismatches has a potential for the contribution to humanities; it will be a powerful tool for a comparative analysis between the transcribed Buddhist sutras and the woodblock-printing ones.

In this paper, we give an introduction to the developing support system for making the documents using the shot images and text files of the sutras, with the aid of the version control software Subversion. Through the experiment, we made sure that the system can be used as a multiuser transcription support tool, but the problem becomes how to integrate the workers' results which might conflicts each other. We then propose the consolidation method as well as the working model for resolving this problem.

2 THE SYSTEM

Kongoji Issaikyo is a collection of thousands of Buddhist sutras which were transcribed nearly 1,000 years ago and now possessed by Kongoji Temple in Japan in good preservation. It is true that the sutra were written in Chinese and currently most Japanese people (and maybe Chinese) who are good command of Chinese characters cannot understand the passages of the sutras, but Buddhism researchers regularly read these historical materials to make clear the religious circumstances of those days. Such researchers consider that the sutras will make a contribution to Buddhist study or historical science of Japan. Although they began to survey the documents decade ago, the researchers has been energetically doing the work about the checkup of the scriptures and the photographing of all the descriptive content. The authors received a number of shot images taken with a digital camera, and since the we have been investigating the automated method for combining the images and existing text files of Buddhist sutras.

In order to search the shot images of Buddhist sutra for a passage, we need the text document which is in strict correspondence with the images. The most established text files of Buddhist sutras are maintained by Chinese Buddhist Electronic Text Association (CBETA,) now. We refer to the data set as CBETA texts. The files are based on Taisho Tripitaka which is derived from Buddhist sutras by woodblock printing. The images of Buddhist scriptures and the CBETA texts in our own hands look largely same, though, we can easily find the difference by character between the image and the text. Some of the differences are due to the transcription error such as a literal error or an omitted error, but some may be the very distinctions that the Buddhism researchers for throwing light on, that is, the clue to the propagation of Buddhist sutras and Buddhism. A goal of our study is to supply a practical support system for comparing the shot images and the relevant texts or for contrasting the edited texts compatible with the Kongoji Issaikyo.

Based on this problem consciousness together with the contents in hand, we are developing the data management system and the interface for enabling a user to read the shot image and the text all together and modify the text so that the text may be compatible with the image. By using the system, we will be able not only to search images through a full-text search of the text files but to get a foothold for learning the difference between transcribed Buddhist sutras and those of woodblock printing. Note that the system are not for correcting the shot image, since we have to show our respect for the historical material. In addition we do not intend to send the modified text to CBETA; we actually attempt to make digital transcriptions of Kongoji Issaikyo or ancient Japanese Buddhist sutras efficiently using CBETA texts.

A typical Buddhist sutra consists of a few thousand Chinese letters. Particular two-letter idioms such as the word meaning bodhisattva appear frequently, and repetitive sequence of letters are often seen. Whether it is transcribed or wood-block printed, the number of letters in line is basically fixed and the characters are arranged in a methodical fashion vertically and horizontally. These properties of the documents imply that it is impossible only for a single worker to finish the absolutely perfect text file. To ensure a quality of the text, we should pay attention to the multiple users' operations.

For supporting the multiple-user modification and management of the documents, we put in use the version control software Subversion (SVN,) in our system. Subversion was originally intended for the version management of source files of software. The tool tells us the difference of the contents between any two points of time, which is suitable for managing and displaying the distinction of the modified text files of Buddhist sutras.

We are introducing several terms around Subversion. A group of files maintained in a server for version control is a *repository*, while a *working copy* is a unit of file manipulation associated with a repository. Note that the repository and the working copy have a one-to-many relationship. An *update* means the operation where the latest version is sent from a repository to a working copy. Conversely the action to transmit the modification within a working copy to the repository is called a *commit* (used as a noun as well as a verb). In doing a commit, the user, *committer*, can leave a message which is commonly called a *com-*

mit log, which will inform the co-workers of why and how the committer went over the files. The change of repository is controlled by a serial number named a *revision*. After a commit, the revision increases by just one.

Subversion is often abbreviated as SVN, while the lower-case word `svn` is a command name for various operations. When executing `svn diff` together with the option about two revisions, we will obtain the line-at-a-time difference of the file between the specified points of time. The file of this output is called a *diff file*. Figure 1 is an example of a diff file, as the result of comparison between revisions 249 and 270. The line with a single minus sign preceding Chinese letters is the one which the text file of revision 249 includes but that of revision 270 does not, and the plus sign means the reverse. The three lines before or after them show the common context. It is easy to know the difference by character, not only with our eyes, but in an automatic way by writing a program.

```
===================================
--- butten.txt (revision 249)
+++ butten.txt (revision 270)
@@ -3,7 +3,7 @@
 比丘眾五千人俱皆是阿羅漢諸漏已盡意
 解無垢眾智自在已了眾事譬如大龍所
 作已辦離於重擔逮得所願三處已盡正解
-已解復有五百比丘尼諸優婆塞優婆夷諸
+已解復有五百比丘尼諸優婆夷塞優婆夷諸
 菩薩摩訶薩已得陀＝尼空行三昧無相無
 願藏已得等忍得無罣礙陀＝尼門悉是五通
 所言柔軟無復懈怠已捨利養無所希望逮
```

Figure 1: Example of diff file.

The typical file management using Subversion separates the disk storage for the repository from the working copies. Various tools for the servers and for the clients are available while the hosting services of the repositories (for example (Unfuddle,)) are offered on the Web. In contrast, our system puts both data in a Web server and invokes the `svn` command within it. Therefore the user can enjoy the services using his or her favorite browser without regard to the working copy and other notions of Subversion.

We briefly explain how to use our system. A user gains access to the top page with a favorite browser, and logs in by entering the user name and the password. If it proves successful, then there is displayed a page for selecting the sutra to read and edit. After that, he or she sees a shot image and a text side by side. On either one of the components, the user can carry out the drag operation so that the both portions of the image and the text may move according to the operation by pixel. Note that the selection using the drag on the text part is prohibited but the smooth scrolling is available instead. There are found several button on the page for jump the display position quickly. After finding a mismatch, he or she can push the button for switching the review mode and the edit mode; in the edit mode, a text form for the horizontal writing is provided to modify the line (Fig. 2). The modified line is highlighted on the text portion (Fig. 3; The second vertical line to the right of the text part grows down a bit.) but the commit is not done yet. When the button for commit is pressed, the modifications are transmitted to the server, and he or she looks at the review mode where there is no highlighted line.

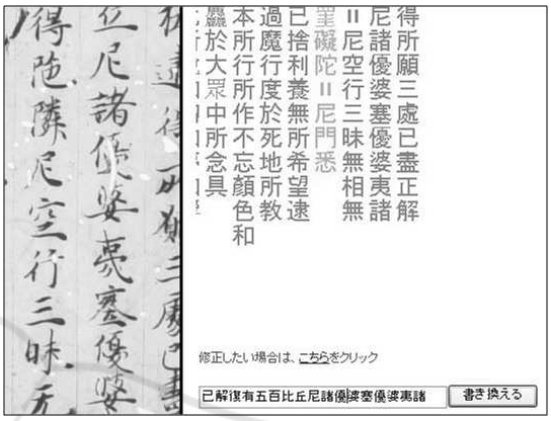

Figure 2: Screenshot before modification.

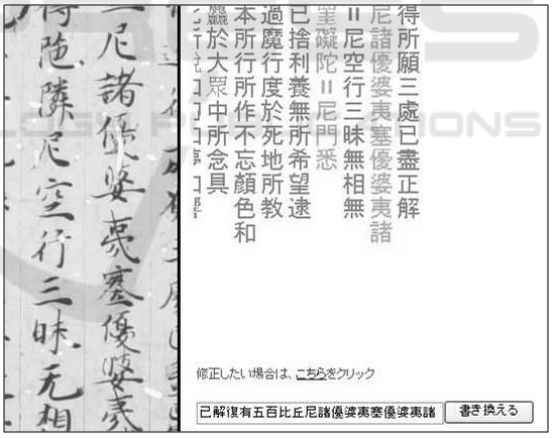

Figure 3: Screenshot after modification.

The following two ways of making use of the system are inside the scope of the assumption.

Auxiliary Function of Review Tool. A user ordinarily enjoys our service as a digital book of Buddhist sutras, but if the divergent characters are found, then he or she can correct the text.

Transcription Support Tool. A worker observes the shot image and the text of a sutra to resolve the mismatch by modifying the text. His or her goal is to make a document where the text is completely coincident with the sutra images.

3 EVALUATION EXPERIMENT

We performed laboratory experiments to verify the usefulness of our system. Five college students put the system in use to make a text file which coincides with the shot image of Kongoji Issaikyo. We got two sutras ready for edit. Any of the sutras has about 450 lines while each line mostly holds 17 Chinese letters. With consideration for the enlisters' health, each person read and edited a sutra for three quarter hour. Prior to the performance, we prepared the text file where the region which does not appear in the corresponding shot image is deleted and the positions of newline character were fitted in with the image.

We would like to discuss how to judge whether each modification is valid or not, under the assumption that there does not exist a correct text which is completely the same as the shot images with regard to the content. A simple way is that users edit a text file simultaneously or during separate intervals. However this naive method has at least two defects. First, when a collision arises, i.e. an attempt of commit overlaps the previously approved commit, it is harder to resolve the collision than a standard file management using Subversion, since the user cannot directly contact with the working copy in our system. The other reason is as to the inequality. For example, after a pioneer finds lots of mismatches and changes them, other co-workers recognize much less discrepancies. In this case, the first worker apparently operate harder according to the commit record. Another possibility attaches importance to later changes; earlier changes might be rewritten again although the last commit will not be examined by the others. Some of the troubles described adobe may be resolved through some sort of control out of our system, but such a constraint might provide an awkward execution environment.

In our experiments, each worker began with the text file which is common if the referred sutra is the same. Actually after a worker ended the reading and the edit, the revision was wound back to that of initial state. (But it is obvious that the rewind would not blot out the previous commits.) It is true that the time-multiplexed operation like this seems ineffective with regard to the experiment and practical use, but we are improving the system so as to permit the concurrent use for a common sutra by introducing branches which are broadly accepted in Subversion-based file managements. After all the operation were finished, we invoked the `svn diff` command to obtain the diff files. As the result of reading the modified contents, we decided that the modified position where at least two persons rewrote is adopted. Furthermore we delivered the adoption judgment for all the modifications by comparing each diff file with the consolidated one.

After the workers had rewritten the mismatched characters using our system for three quarter hour per sutra, we processed their achievements by means of the above procedure. As seen in Table 1 (Fukuoka et al., 2009), the number of examined lines ranged 33 to 109, the number of modification 8 to 26, and the number of contribution 3 to 11. There does not exist any collision having need of choice where a character is replaced by different ones. When we cast a spotlight on the consolidated text files, there were found 41 and 43 modifications in the respective sutras which someone had made, and 28 and 14 modifications were done by two workers or more.

Table 1: Workers' activities.

Sutra	Worker	Lines	Mods	Adopts
1	A	70	8	4
1	B	50	18	11
1	C	69	13	4
1	D	36	14	5
1	E	38	9	4
2	A	109	26	10
2	B	64	10	5
2	C	75	8	6
2	D	33	16	3
2	E	96	9	8

Lines: Number of examined lines.
Mods: Number of modifications.
Adopts: Number of adoptions.

4 INTEGRATION OF THE MODIFICATION WORKS

In this section, we formulate the process where the consolidated version of text is constructed out of the diff files of the workers.

In advance, an unrevised text file is split into characters to which identification codes are assigned respectively. The straightforward way would be to number the characters serially, and the rest of this paper adopts it. We can instead use a pair of the line number and the position of the line, as long as the values are ordered along with the order of characters. For a valid integer p, the character which corresponds to the number p according to the above numbering is called "the character at p". By using this coding, we formulate the modification of a worker.

If a worker insert a character α just after the character at p, then the action is denoted as $p+\alpha$. In the case of prepending a character α to the text, $0+\alpha$ is

the code. Putting a string say $\alpha_1\alpha_1\cdots\alpha_n$ in right after the character at p is associated with $p+\alpha_1\alpha_2\cdots\alpha_n$.

The deletion of the character at p is written as $p-$.

We present the form of the substitution as well, instead of combining the insertion and the deletion. If the character at p is replaced by a sequence $\beta_1\beta_2\cdots\beta_m$, then the code is $p!\beta_1\beta_2\cdots\beta_m$. Typically the substituted character is only one say β, and then $p!\beta$ is the desired word. The instruction where the string of n characters is replaced by that of m characters is represented as a replacement word and $m-1$ deletion words. Moreover we define a *correction word* in terms of the word formed by a number, a symbol which is one of the plus sign, the minus sign or the exclamation mark, and a string which is empty following the minus sign.

A given diff file is transformed into the sequence of the correction words. To cope with complicated differences, we may deploy the algorithm for calculating the Levenshtein distance (Gusfield, 1997). All the correction words are sorted by the position, the operator and the trailing string. Once extracting the list which is called the *correction array*, we can make the resulting text file of a worker from the original text file and the correction array; all we have to do is to apply each correction word of the array in reverse. Finally the correction array is the essence of the differential with regard to the character, whereas the diff file indicates the information of lines.

Now we explain how to consolidate the workers' fruits through their correction arrays (see Fig. 4). Let N be the number of the workers, and W_i stand for the correction array for the i-th worker. The next step is to configure the array W by sorting the union of W_i for all $1 \leq i \leq N$. Assume that W_i and W can be regarded both as the sets and as the arrays, i.e. the judgment of belongingness and the random access with an index are available to the discrete or consolidated correction array. However W should not be the set that we would like to obtain, since W may include inadequate instructions. In addition, there might exist a pair of correction words (called *collision words*) with a common position and different instructions of insertion, deletion or displacement, in W, although W_i does not hold such a couple. We denote $\#W$ to represent the number of correction words in W and $W[j]$ $(1 \leq j \leq \#W)$ to mean the j-th word of the array W.

The matrix A is composed which has N rows and $\#W$ columns. The component A_{ij} is determined by comparing W with W_i, that is:

$$A_{ij} = 1 \quad \text{if } W[j] \in W_i. \tag{1}$$

$$A_{ij} = 0 \quad \text{if } W[j] \notin W_i. \tag{2}$$

By definition, each column has at least one 1 component. If the number of ones is larger than a threshold which may be a constant or a half of the workers, then the modification corresponding to the column is accepted. When a small threshold is employed, collision words may be accepted, and then either one should be selected in some fashion. Finally we can obtain the matrix A' whose columns are the accepted ones and the valid correction array W' composed of the accepted correction words.

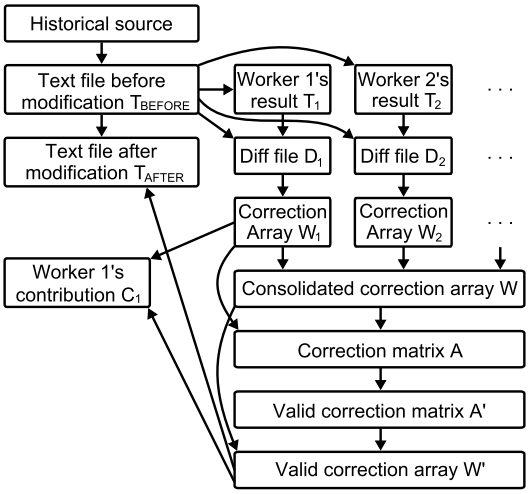

Figure 4: Workflow of integrating workers' modification.

When applying the correction word of W' reversely to the text file before modification, we will get hold of the text file where the workers' results are consolidated.

We can quantify the i-th worker's contribution by comparing W_i with W'. In concrete terms, $\#(W_i \cap W')$ means the number of worker i's modifications which are instrumental in making the resulting text file, or the adoption number. The rejection number is derived by $\#(W_i - W')$ where the minus sign denotes the set subtraction. These numbers can convert into the ratios using the division by $\#W_i$, which may be more suitable for evaluating the worker's performance.

To make sure that this formulation works well, we apply it to a simplified example of a string manipulation. Let T_{BEFORE} be "japanese", and assume that three workers have changed the sequence of characters into "jopanés", "japonés" and "japonese" respectively. The values of the strings, arrays and matrices according to the above formulation are as follows:

$$T_{BEFORE} = \text{“japanese”}. \tag{3}$$

$$T_1 = \text{“jopanés”}. \tag{4}$$

$$T_2 = \text{“japonés”}. \tag{5}$$

$$T_3 = \text{“japonese”}. \tag{6}$$

(D_1–D_3 are omitted since they are no use in this case.)

$$W_1 = \{2!\text{o}, 6!\text{é}, 8-\}. \tag{7}$$

$$W_2 = \{4!o, 6!é, 8-\}. \quad (8)$$

$$W_3 = \{4!o\}. \quad (9)$$

$$W = \{2!o, 4!o, 6!é, 8-\}. \quad (10)$$

$$A = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (11)$$

$$A' = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}. \quad (12)$$

$$W' = \{4!o, 6!é, 8-\}. \quad (13)$$

By applying the correction words in W' to T_{BEFORE} in reverse, via "japanes" and "japanés", we finally obtain the string "japonés" which is assigned to T_{AFTER}.

Let the contribution be the number of ones of the row of A' which corresponds to the worker (equal to the adoption number), then:

$$C_1 = 2. \quad (14)$$

$$C_2 = 3. \quad (15)$$

$$C_3 = 1. \quad (16)$$

In the case of evaluation experiment in the previous section, the column of Adopts in Table 1 is equal to the contribution mentioned just above. When dividing the values by that in the column of Mods respectively, we can obtain the ratios of adoption. The ratios varied from 3/16 to 8/9.

5 CONCLUSIONS

In the anterior half of this paper, we have introduced the documentation support system which utilizes Subversion and enables one to read the shot image and the text at a time. The second half was devoted to provide the workflow of consolidating text file from the separated achievements of the workers. The more workers are engaged in the revision, the more mismatches will be detected, but we have no idea of the quantitative relationship among the numbers of workers, mismatches, and what should be corrected. We will have to conduct experiments from this viewpoint, as well as continue to improve the interface for practical use by the researchers of Buddhism or historical science.

This application is able to manage text files by comparing the files with the image files of copytext, no matter what language the copytext was written in. We request that the text should consist of Unicode characters, since the system is obliged to output diff files where UTF-8 is used as the character code. If you would like to express the character outside this encoding, then it should be specified as a string which is regarded as a character in going for the correction word, using some sort of escape sequence.

We developed the documentation support system under the assumption that the text files which resemble the shot images of sutra in content are available. For applying to other sorts of materials, it would be one way to make T_{BEFORE} with a character sensing equipment which may not be accurate enough.

REFERENCES

Alshuhri, S. S. (2008). Arabic manuscripts in a digital library context. In *11th International Conference on Asian Digital Libraries (ICADL 2008)*, LNCS 5362, pages 387–393.

CBETA. Chinese Buddhist Electronic Text Association (CBETA). http://www.cbeta.org/. (in Chinese).

Chen, S.-P., Hsiang, J., Tu, H.-C., and Wu, M. (2007). On building a full-text digital library of historical documents. In *10th International Conference on Asian Digital Libraries (ICADL 2007)*, LNCS 4822, pages 49–60.

EPOCH (2005). Report on common research agenda.

Fukuoka, H., Murakawa, T., Noda, D., and Nakagawa, M. (2009). Evaluation of support system for revising Buddhist canons using subversion. In *IPSJ Symposium*, Vol.2009, No.16, pages 61–66. (in Japanese).

Gusfield, D. (1997). *Algorithms on strings, trees, and sequences: Computer science and computational biology*. Cambridge University Press.

Ribeiro, C., David, G., and Calistru, C. (2007). Multimedia in cultural heritage collections: A model and applications. In *10th International Conference on Asian Digital Libraries (ICADL 2007)*, LNCS 4822, pages 186–195.

SVN. Subversion. http://subversion.tigris.org/.

Tanaka, T., Nino, Y., Zhang, R., Rolland, M., Nakagawa, M., Aoki, S., Utsunomiya, K., and Ochiai, T. (2006). A database system of Buddhist canons. In *Seventh Joint Conference on Knowledge-Based Software Engineering*, pages 327–336.

Unfuddle. http://unfuddle.com/.