2009

Chinese localisation of Evergreen: an open source integrated library system

Qing Zou

Liu Guoying
University of Windsor

Chinese localisation of Evergreen: an open source integrated library system

Qing Zou and Guoying Liu

Authors: Qing Zou is Systems Librarian at Lakehead University Library, Ontario, Canada. E-mail: qzou@lakeheadu.ca

Guoying Liu is Systems Librarian at University of Windsor Leddy Library, Ontario, Canada.

Abstract

Purpose – To investigate various issues related to Chinese language localisation in Evergreen, an open source integrated library system (ILS).

Design/methodology/approach – A Simplified Chinese version of Evergreen was implemented and tested and various issues such as encoding, indexing, searching, and sorting specifically associated with Simplified Chinese language were investigated.

Findings – Unicode eases a lot of ILS development problems. However, having another language version of an ILS does not simply require the translation from one language to another. Indexing, searching, sorting and other locale related issues should be tackled not only language by language, but locale by locale.

Practical implications – Most of the issues that have arisen during this project will be found with other ILS-like systems.

Originality/value – This paper provides insights into issues of, and various solutions to, indexing, searching, and sorting in the Chinese language in an ILS. These issues and the solutions may be applicable to other digital library systems such as institutional repositories.

Keywords – Localisation**,** Evergreen, Simplified Chinese, ILS, Open source software, Library management systems, Canada

Paper type – Case study

Word count: 5,022

1. Introduction

Evergreen Open Source integrated library system (ILS) is a free, open source library management system developed and currently in use by Georgia Library Public Information Network for Electronic Services (http://open-ils.org/). Evergreen has attracted worldwide interest since its introduction in 2006. It is used at Georgia Public Library System, known as PINES – Public Information Network for Electronic Services, of some 275 public libraries throughout the state. Other examples of users include 16-strong Sitka consortium of public libraries in British Columbia, Canada and the Robertson Library at the University of Prince Edward Island, also in Canada. In July 2007, the University of Windsor, McMaster University, and Laurentian/Algoma Universities, in Ontario, Canada, initiated the collaborative project Conifer which aims to share a common instance of Evergreen. The target date for Conifer going live with Evergreen is May 2009. (Rhyno, 2008) This triggers the interest of the authors of this paper from the University of Windsor and Lakehead University to work on an Evergreen Chinese version.

A full-fledged ILS not only provides capabilities to display but also to sort, index, and search materials in different languages. Evergreen is still a new ILS in comparison with other commercial systems. It was originally developed in North America and best supports English. The localisation for other languages is a key project for its promotion to non-English speaking countries or libraries with non-English collections. So far, three languages other than English are under development including French, Spanish, and Simplified Chinese versions. The first Chinese version was released in 2008 and has received vast interest from many libraries all over the world, including the Sitka consortium in British Columbia, Canada, and the China Evergreen

Rural Library Service (CERLS), a nationwide program in mainland China built by Evergreen Education Foundation (http://www.evergreeneducation.org/).

In terms of library system localisation, Asian languages are very different from, and much more difficult than, Latin-based languages. During the development of the Simplified Chinese version of Evergreen, we encountered a variety of issues associated with the Chinese language in either English or Chinese predominated environments. This paper aims to investigate these issues as well as to examine the solutions. The findings will have significant implications for library systems intending to support Chinese and other languages.

2. Literature review

Many studies have been done on the software localisation or globalisation, but few on library systems, especially on an open source ILS. Carey's (1998) study spells out the general issues of translation, organisation, culture, interface design and documentation for creating global software. Collins (2001) addresses the technical issues, such as language, currency, formats, measurements, and collating sequences, in software product and interface localisation. Singh, Zhao and Hu (2003) conducted a study on the cultural values to web content through a comparative analysis of the US-based companies' domestic websites and their Chinese websites. Guo (2003) examined technical and cultural issues along with the text translation. Some of these studies (Carey, 1998; Collins, 2001) point out the problems with Chinese character support and the impact of Unicode, but do not further examine the issues with encoding, indexing, searching and sorting, which are crucial to library systems.

In libraries in North America, the Chinese characters are generally encoded in special fields (880s) in MARC21, while romanised form still occupies the primary descriptive fields for

authors and titles (Coyle, 2006). Wade-Giles has been a *de facto* standard for Chinese romanisation in libraries for decades. Since 2000, the Library of Congress (LC) has moved to the Pinyin system and converted all its Chinese records into Pinyin (Library of Congress, 2005). At present, the Pinyin system has replaced Wade-Giles in most libraries in the world. In written Chinese, there are about 47,000 to 85,000 distinct characters sharing only around 1,300 syllables, including tonal distinctions, in spoken Chinese (Arsenault, 2001). To reduce the homophonous ambiguity, the normal practice with Pinyin romanisation is to separate words from each other with a space. The word division and syllable aggregation, however, becomes an issue in the retrieval of Chinese materials (Huang, 2004). Further, Pinyin is a standard based on one of northern dialects of China. Even Chinese native speakers have difficulties in converting Chinese into Pinyin correctly. In the cataloguing process, it is impossible to keep consistency when converting Chinese into Pinyin and even more difficult to aggregate syllables into lexical units. In the meantime, users may form queries differently using aggregation or not. In addition, it was estimated that only 4,000-7,000 characters in Chinese vocabulary are frequently used. For those infrequently used characters, people may not even know the pronunciation and thus cannot access them by phonetic scripts (Shi and Larson, 1989).

The value of producing records of Chinese materials in anything other than the original form is limited. Original script is obviously the basis of the record in modern Eastern Asia Machine Readable Cataloging (MARC) (Helliwell, 2006). Chinese MARC (CNMARC) was created by the National Library of Beijing in 1986 based on the UNIMARC standard. Since 1989, CNMARC has been set up as the standard for Chinese materials while USMARC is used for Western language materials in mainland China (Liu and Shen, 2002). In Taiwan, Chinese MARC (CMARC) was published in 1982 and is still used by most libraries there (Mao and Hsu,

2007). In Hong Kong, the academic libraries have a long history of following MARC21 in cataloguing (Pong and Cheung, 2006).

In Western countries, a few library automation projects have addressed Chinese character support since the 1980s. OCLC took its first step to automating CJK (Chinese, Japanese, and Korean) languages. It offers derived search keys based on romanisation as well as CJK search keys for title, name, etc. (Wang, 1985). At present, both OCLC and RLIN already offer the optional feature of providing software support for reading Chinese characters (Huang, 2004).

Computer processing of Chinese characters, including encoding, indexing, searching and sorting, has been studied since the 1970s (Wu and White, 1990). Coding Chinese script into computer is not a big problem any more. However, the variety of Chinese encoding standards used in different libraries or areas, for example, GB in mainland China, Big5 in Taiwan and Hong Kong, hinders data exchange among library systems (Pong and Cheung, 2006).

Unicode is an industry standard for computers to encode the characters in the scripts of the world's languages. With the development of Unicode, major players in the information industry support Unicode at all levels. Windows Operating Systems (OS) and Mac OS support Unicode at the system level for character and string manipulation. Support for Unicode in Linux has been addressed in different Linux implementations. The Linux Internationalisation Initiative defined a globalisation specification, OpenI18N, released in 2002 that includes support for Unicode (Tull and Straley, 2003). Database vendors like Oracle, Sybase and IBM are among the companies involved in developing Unicode. Their database products are fully compatible with Unicode. Open source database software like MySQL and PostgreSQL are also compatible with Unicode.. At the application level, web browsers, such as Internet Explorer, FireFox, and Safari, have the capabilities to decode Unicode-encoded web pages. All these developments and

advancements have had great impacts on the library world. Technologies have simplified the process of inputting, displaying, indexing, and searching Unicode encoded data.

According to Tull's (2002) survey on the Unicode implementation within ILSs, almost all of the responding library system vendors "had either implemented Unicode or had plans to implement Unicode by the end of 2003". Abram and Huang (2007) described the large ILS company, SirsiDynix's experience with the adoption of Unicode and the Unicode issues with CJK. They discussed the issues of Chinese sorting, indexing and searching in library systems through a case study on the Horizon ILS. Different sorting rules are adopted in different areas. For example, mainland China uses Pinyin, Taiwan and Hong Kong use the 'stroke-radical'. Because of the nature of the Chinese language, there is no obvious boundary between words in Chinese text except for punctuation symbols in a sentence. Various approaches, including character-based and word-based word segmentation, have been proposed and studied regarding this issue (Foo and Li, 2002; Nie and Ren, 1999; Zhou, 2001). However, due to the complexity of the Chinese language, there is no one single approach that has been adopted as the *de facto* standard for information systems.

3. The architecture of Evergreen

> "Evergreen is an enterprise-class library automation system that helps library patrons find library materials, and helps libraries manage, catalog, and circulate those materials, no matter how large or complex the libraries… Evergreen is open source software, freely licensed under the GNU GPL" (http://open-ils.org/).

As an open source ILS, Evergreen has a very unique system architecture utilising various open source projects. Its server-side services, which have been developed in the computer languages C and Perl, are handling all kinds of requests including authentication, storage and circulation. “Messaging is an important part of application frameworks, allowing different processes to exchange information” (Weber, 2006, p.40). Evergreen’s unique message core is called open scalable request framework (OpenSRF) built on Ejabber, an open source real time instant messaging service. The OpenSRF not only provides secure message passing and scalable architecture, but a simple application programming interface (API) for application developers. In particular, the OpenSRF Router provides failover and load balancing (Singleton and Erickson, 2006). Server services or application servers connect to the Ejabber server and register with the router. All requests for a given service or an application server will be directed to the jabber router and routed to the next server process. OpenSRF sessions will be opened for ‘stateful’ communication.

Extended Perl and C modules for the Apache web server have been designed for the public interface presentation layer. All requests from the public interface are formed in the JavaScript Object Notation (JSON) format and then directed to the proper server service. XML formatted result data are returned. However, the Apache C modules interpret XML data into XHTML pages, which modern browsers are able to understand and render them for users.
In the database layer, Evergreen utilises the powerful open source database management software, PostgreSQL. The storage service connects to the database, sends search requests to the database, and receives search results from the database. Evergreen makes use of the Tsearch2 of PostgreSQL, which will be discussed in detail later. The staff client, built on the XML user interface language (XUL), can easily be set up on Windows and Mac X operating systems and is

not limited to Linux (http://open-ils.org/).

4. Chinese localisation of Evergreen

Localisation is the translation and adaptation of a software or web product, which includes the software application itself and all related product documentation (Esselink, 2000).

Except for character encoding, the following list of cultural conventions need to be considered for purpose of localisation:

Address Format	Measurements
Collation/Sorting rules	Messages
Currency	Name Format
Dates/Times	Numbers/Telephone numbers

Usually, the cultural conventions of a country are termed in the country's locale. Also the following features of Evergreen are crucial to the process of localisation:

- The server side of Evergreen runs on Linux
- Evergreen fully supports Unicode Transformation Format 8 (UTF-8)
- Evergreen leverages the Tsearch2 for indexing and searching
- Evergreen stores records in MARC XML format

Along with the development of Operating Systems, especially Linux, and the creation of GNU gettext and many other translation projects, it is not difficult to display or input Chinese characters any more. It is neither an issue for browsers to display Chinese characters nor for databases to store Chinese characters. Although there is a variety of issues with localisation, our focus is on the following four major issues: interface localization; indexing; searching; and

sorting in the Chinese localisation of Evergreen version 1.2.0.

4.1 Interface localisation

Although translation is just one aspect of localisation, no doubt it is very important for every localisation project. Translation is a process mainly dealing with messages and documentation in a system. There are many open source tools available to alleviate the technical burden of the translation process. Often, translators are not technologically ‘savvy’ people. Co-operation among translators is also a key to making the translation consistent and to having quality output. Specifically, for this project, a Pootle server, which is a user-friendly web translation management tool, was set up (http://translate.sourceforge.net/wiki/pootle/index?redirect=1). It allows translators to log in and to translate one or more messages from a source language to a target language, as can be seen in Figure 1.

Take in Figure 1

However, the Pootle server manages Portable Object (PO) files and translation output is stored into PO files directly. In Evergreen, the messages of its public access interface are largely controlled by two Document Type Definition (DTD) files. Linux utilities such as po2moz were utilised for converting PO files into DTD files. Furthermore, a ‘cron’ job has been created to convert PO files into DTD files every hour. This means that the messages in the public access interface will be updated hourly. The default locale of Evergreen is American English (en-US). To comply with locale conventions, the zh-CN locale has been created in Evergreen for Simplified Chinese. Also the two-hourly updated DTD files in Chinese have been included into the newly created zh-CN locale. By changing the environment variable controlling which locale to use, Evergreen will be able to have a public access interface in Simplified Chinese. Figure 2 shows the search interface of Chinese version of Evergreen. A similar process can be applied to

Traditional Chinese with locale name zh-TW.

Take in Figure 2

4.2 Indexing in Evergreen

Tsearch2 is a Full Text Search (FTS) engine for PostgreSQL, and is fully integrated into PostgreSQL 8.3. A typical searching string will be indexed by words (Rhodes, 2003). FTS uses the four tables to configure and control itself: pg_ts_dict; pg_ts_parser; pg_ts_cfg; and pg_ts_cfgmap. However, FTS is an addition to PostgreSQL. Built-in functionalities of PostgreSQL could not handle and support FTS. The ‘to_tsvector’ function of FTS is used for full-text indexing. For example, a title like “Social history of China” will be separated and indexed as following:

```
# SELECT to_tsvector('Social history of China')

          to_tsvector
-----------------------------------
'china':4 'social':1 'histori':2
(1 row)
```

FTS can be considered to utilise a unigram word-based algorithm. Unigram is a special case of n-grams algorithm (known as sequences of consecutive words) when n equals one. Word-based n-grams can be considered as a kind of window of n words in size progressively sliding along the words of documents (Figuerola, et al. 2000). Thus, for instance, the title “social history of china” would give rise to the following unigrams: ‘social’, ‘history’, ‘of’, and ‘china’. However, FTS has done some extra work. More strictly, the algorithm used in indexing is similar to the inverted index. FTS also has the ability to filter out stop words like 'of'. Stemming algorithms have been used to index 'history' as 'histori'.

The ‘to_tsquery’ is the counterpart function for indexing. Similarly, the query as

follows is used for searching records having words “social” and “history”.

```
# SELECT to_tsquery('default', 'social&history')
```

Evergreen leverages FTS for indexing. All FTS indexing in Evergreen is through a database trigger. Therefore, when either updating records or inserting records, the database trigger will be invoked to index data into an FTS-understandable format. The following is the original definition of the ‘oils_tsearch2’ trigger:

```
CREATE OR REPLACE FUNCTION oils_tsearch2 () RETURNS TRIGGER AS $$
BEGIN
   NEW.index_vector = to_tsvector(TG_ARGV[0], NEW.value);
   RETURN NEW;
END;
$$ LANGUAGE PLPGSQL;
```

This trigger simply converts regular database fields into the FTS indexing format utilising the ‘to_tsvector’ function.

However, this is not enough to handle Chinese in Evergreen for two reasons. First, by default, the en-US locale has been installed and set up with the ‘pg_ts_cfg’, ‘pg_ts_dict’, and ‘pg_ts_cfgmap’. Therefore, in order to utilise the FTS for handling Chinese text, some configuration data must be added into the three tables. More importantly, FTS relies on spaces between words to index words. However, there is no obvious boundary such as a space between words in Chinese. Therefore, a specific algorithm for Chinese must be incorporated into FTS for segmenting texts into words or Chinese characters. For demonstration purposes, the simple unigram algorithm is selected as the algorithm for indexing and searching in Evergreen. This algorithm is to add spaces between Chinese characters as word boundaries in English. Then the

segmented Chinese text can be understood and handled as text in English using the ‘to_tsvector’ function. The following is a unigram implementation on PostgreSQL database using its PL/PGSQL language:

```
CREATE OR REPLACE FUNCTION unigram(input text) returns text as $$ ...
BEGIN
  query := lower(regexp_replace(input,'[[:punct:]]',' ','g'));
  i := char_length(query);
  LOOP
    thisVal := substring(query from j for 1);
    IF ((thisVal <> ' ')AND(thisVal <> '    ')) THEN
      IF (is_chinese(lastVal) OR is_chinese(thisVal))  THEN
        retVal := retVal || thisVal;
      ELSE
        retVal := retVal || ' ' || thisVal;
      END IF;
    END IF;
    lastVal := thisVal;  j := j+1;
    EXIT WHEN j > i;
  END LOOP;
  RETURN trim(retVal);
END
$$language plpgsql;
```

This unigram function simply adds spaces between Chinese characters and keeps non-Chinese

characters intact. For instance, the following Structured Query Language (SQL) query utilizing this function will be able to separate the “中国社会□史” into “中 国 社 会 □ 史”.

```
# SELECT unigram('中国社会□史');
          unigram
------------------------------------
'中 国 社 会 □ 史'
(1 row)
```

By simply applying this to the database trigger for FTS indexing in Evergreen, Evergreen will have the ability to index Chinese text. The modified ‘oils_tsearch2’ trigger in Evergreen is listed as follows:

```
CREATE OR REPLACE FUNCTION oils_tsearch2 () RETURNS TRIGGER AS $$
BEGIN
   NEW.index_vector = to_tsvector(TG_ARGV[0],coalesce(unigram(NEW.value),
'));
   RETURN NEW;
END;
$$ LANGUAGE PLPGSQL;
```

The ‘to_tsvector’ function of FTS will be able to index title like “中国社会历史” as “中”, “国”, “社”, “会”, “历”, and “史”. This means that this six Chinese characters phrase is linking to the six separate Chinese characters. Searching one or any of the six Chinese characters will be able to bring up this title.

4.3 Searching in Evergreen

To some extent, searching is the opposite process of indexing. In FTS, a typical search query will

be divided into word divisions, which are especially important for handling Chinese language. For example, a query “Social history of China” will be segmented into “social”, “history”, and “china” to search against database in Evergreen. Similarly, a query “中国社会历史” (social history of China) needs to be segmented as “中”, “国”, “社”, “会”, “历”, and “史” using the same algorithm as we used for indexing. The rational of using the same algorithm in both index and search stages is to maximise the amount of results with reasonable recall and precision. In Evergreen, before a search query passes into the module for real database search, it will be converted into searchable terms “social&history&china” similar to the process of segmenting “social history of China”. For the search query “中国社会历史”, it will be formed as “中&国&社&会&历&史”. Once the query is formed in the above format, FTS will understand and try to find records having the Chinese characters: “中”, “国”, “社”, “会”, “历”, and “史”. Then the transformed terms will be incorporated into real database SQL queries. A SQL query of a keyword search for “中国社会历史” will be similar to the following simplified SQL query:

```
SELECT b.id,
  AVG( (rank(keyword.index_vector, to_tsquery('keyword','中&国&社&会&历&史'))
AS rank,  b.source
FROM metabib.keyword_field_entry keyword, metabib.rec_descriptor rd,
                config.bib_source src, biblio.record_entry b
WHERE
  keyword.index_vector @@ to_tsquery('keyword','中&国&社&会&历&史') AND
  b.id = keyword.source
GROUP BY b.id, b.source
```

If the search query includes any of the following searches for “中国” (china), “社会” (social), or “历史” (history), or even the meaningless phrase of “中历”, the system will return the record having “中国社会历史”. This results in perfect recall and having lower precision.

As mentioned above, queries need to be submitted as ‘<word>&<word>…<word>’ format in Evergreen. A logical solution is to apply an algorithm which can segment Chinese into the above format right before search queries are translated into SQL queries. In Evergreen, ‘FTS.pm’ and ‘fts.pm’ are two Perl packages used to compose queries into Tsearch2 powered SQL queries. By modifying the ‘fts.pm’ and adding the algorithm, queries in Chinese can be as understandable in Evergreen as queries in English. Therefore, queries in either English or Chinese can be treated as same as each other in terms of searching. Searching Chinese can be achieved without any major structural changes in Evergreen.

4.4 Sorting issues in Evergreen

Because of the nature of the Chinese language, sorting in Chinese is totally different from English. There is no such thing as alphabetic order in Chinese. However, this does not mean that there are no sorting mechanisms for the Chinese language. For example, Pinyin is a very popular sorting method used in mainland China. Basically, Pinyin uses the Roman alphabet to represent pronunciations of Chinese characters. For instance, “zhong guo she hui li shi” is the Pinyin for “中国社会历史”. As a direct result of romanisation, sorting Chinese characters can be achieved by sorting the corresponding romanised alphabet. For many reasons, Pinyin might be the easiest way to sort Chinese in computer systems.

In Evergreen, by default, the system sorts in alphabetic order, more precisely, in the internal code order of UTF-8. The defined characters of ISO-8859-1 are almost identical to the first 128 characters of Unicode and its transformation format UTF-8.In Unicode, the default

sorting method of Chinese characters, is Radical Stroke. However, this sorting method is rarely used by the Chinese. Therefore, this system default sorting is useless for sorting Chinese records, especially for records in Simplified Chinese.

As mentioned above, Pinyin might be an easy way to get records in Chinese sorted in Evergreen. In PostgreSQL, Chinese characters in the UTF-8 format will not automatically be mapped into Pinyin. A proposed solution is to create extra functionalities for mapping Chinese characters into Pinyin on the fly when sorting happens.

4.5 Importing Chinese language materials into Evergreen

Libraries in North America use MARC21 to handle Chinese language materials. Evergreen has the capability to bulk load MARC records either in MARC21 or MARC XML format. However, in mainland China, libraries are using Chinese MARC (CNMARC). Although CNMARC is similar to MARC21 and has been developed based on UNIMARC, there are some distinct differences that prevent us from using the Evergreen MARC importing utility. In CNMARC, the 100 field is heavily used for different purposes. Also, $9 subfields have been widely used in CNMARC. More importantly, Chinese characters in most CNMARC records are encoded in GuoBiao (GB – National Standard), a mandatory standard for all computer applications used in mainland China. The development of GB18030, of which GB18030-2005 is the latest standard for Chinese coded character set, is closely related to the process of developing ISO 10646 and Unicode. A one-to-one mapping can be drawn from GB18030 to ISO 10646. Therefore, Chinese characters encoded in GB18030 can easily be converted into UTF-8. The importing utility of Evergreen can handle MARC21 encoded in UTF-8. So a new importing utility for loading CNMARC can be created by adding conversion and mapping from CNMARC to MARC21.

5. Discussion

For indexing and searching Chinese, only the simple unigram, overlapping bigram, or unigram combined with overlapping bigram, or other similar algorithms are implemented in the ILS. Those algorithms are easy to implement and yield reasonable results with satisfactory system performance. Index and search functions might be optimised to use advanced algorithms for better performance. Some algorithms based on the dictionary approach could potentially lead to better search results. It is, however, not trivial to maintain a dynamic dictionary. Adopting any of these advanced algorithms would introduce more complexity into the system.

Although Traditional Chinese is different from Simplified Chinese in many ways, the indexing, searching and sorting solutions mentioned above are applicable to Traditional Chinese. As one of the benefits of using Unicode, the searching, indexing, and Pinyin sorting algorithms can directly be applied to Traditional Chinese without any adjustment. However, Pinyin might not be in compliant with sorting conventions for people using Traditional Chinese. Hence, different sorting methods particularly for Traditional Chinese need to be investigated.

There are various approaches to tackle the sorting issues in Chinese. Some solutions require changes to the Evergreen database scheme and the SQLs used in search modules. Currently, our proposed Pinyin mapping algorithm will romanise Chinese characters based on a word's most popular pronunciation. An obvious and major issue with this approach is the polyphonic problem which is very common in Chinese language. More accurate algorithms might need to be implemented for better sorting results.

Fonts are a potential issue. The Arial Unicode font from Microsoft might be the best font so far. However, it includes only partial CJK Unicode, and lacks CJK Unified Ideographs Extension B (U+20000 through U+2A6D6). For libraries, it might not be extensive enough to

handle and display rare book records.

In addition to the above-mentioned issues, a couple of other issues needs to be considered in Chinese localisation as well, for examples, staff client localisation, including locale related date, name, and address formatting issues.

6. Conclusion

No doubt, Unicode eases many of the problems of handling different languages. However, Unicode is not the answer for everything. Unicode has partially solved the problems associated with localisation. Different languages and locales will need to be addressed differently in a system, especially in an entire integrated library system. It seems that it is more practical to tweak a system language by language.

References (All URLs were checked 14th September 2008)

Abram, S.and Huang, L. (2007), "Unicode: OPACs and portals, one vendor's experience", *International Cataloguing and Bibliographic Control*, Vol. 36 No.3, pp. 61-64.

Arsenault, C. (2001), "Word division in the transcription of Chinese script in the title fields of bibliographic records", *Cataloging and Classification Quarterly*, Vol. 32 No. 3, pp. 109-137.

Carey, J. M. (1998), "Creating global software: a conspectus and review", *Interacting with Computers*, Vol. 9 No. 4, pp. 449-465.

Collins, R. W. (2001), "Software localization: issues and methods", The 9th European Conference on Information Systems, Bled, Slovenia, June 27-29, 2001.

Coyle, K. (2006), "Managing technology – Unicode: the universal character set, part 2: Unicode in library systems", *Journal of Academic Librarianship*, Vol. 32 No. 1, pp. 101-103.

Esselink, B. (2000), *A Practical Guide to Localization*, John Benjamins, Amsterdam and Philadelphia.

Foo, S. and Li, H. (2004), “Chinese word segment and its effect on information retrieval”, *Information Processing and Management*, Vol. 40 No. 1, pp.161-190.

Guo, S., (2003), “Learning from software localization”, *British Journal of Educational Technology*, Vol. 34 No. 3, pp. 372-374.

Helliwell, D., (2006), “East Asian cataloguing in British libraries: the contribution of Allegro”, *Catalogue & Index*, No. 153, pp. 4-6.

Huang, J. (2004), “Retrieval of Chinese language titles in Pinyin: a comparative study”, *Information Technology and Libraries*, Vol. 23 No. 3, pp. 95-100.

Library of Congress (2005), “Library of Congress Pinyin conversion project”. Available at: http://www.loc.gov/catdir/pinyin/pinyin.html

Liu, S., Shen, Z. (2002), “The development of cataloging in China”, *Cataloging and Classification Quarterly*, Vol. 35 No. 1/2, pp. 137-154.

Mao, C. A. and Hsu, C. F. (2007), “Chinese MARC (Taiwan) and its bibliographic database”. *International Cataloguing and Bibliographic Control*, Vol. 36 No.3, pp. 58-60.

Nie, J.and Ren, F. (1999), “Chinese information retrieval: using characters or words?”, *Information Processing and Management*, Vol. 35 No. 4, pp. 443-462.

Pong, J. and Cheung, C. (2006), “Cataloging of Chinese language materials in the digital era: the cataloging standards and practices in China, Taiwan and Hong Kong”, *Journal of Library and Information Science*, Vol. 32 No. 1, pp.53-65.

Rhodes, B. C. (2003), *Tsearch2 Guide*. Available at http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/docs/tsearch2-guide.html

Rhyno, A. (2008), “Project Conifer”. Available at: http://www.osbr.ca/ojs/index.php/osbr/article/view/691/657

Shi, Y. and Larson, R. (1989), “Facilitating Chinese character entry and information retrieval through regular expression searching”, *Library and Information Science Research*, Vol.11, pp. 335-355.

Singh, N., Zhao, H. and Hu, X. (2003), “Cultural adaptation on the Web: a study of American companies’ domestic and Chinese websites”, *Journal of Global Information Management*, Vol. 11 No. 3, pp. 63-80.

Singleton D., Erickson B. (2006), *PINES Presentation*. Available at: http://www.usg.edu/events/gold/2006/walker_evergreen.pdf

Tull, L. (2002), “Library systems and Unicode: a review of the current state of development”, *Information Technology and Libraries*, Vol. 21 No. 4, pp.181-185.

Tull, L. and Straley, D. (2003), “Unicode: support for multiple languages at the Ohio State University Libraries”, *Library Hi Tech*, Vol. 21,No. 4, pp. 440-450.

Wang, A. H. (1985), “OCLC’s cataloging capability in the Chinese, Japanese, and Korean languages”, *Journal of Educational Media and Library Science*, Vol. 23 No. 1, pp. 57-62.

Weber, J. (2006), “Evergreen: your homegrown ILS”, *Library Journal*, Vol. 131, Issue 20, pp. 38-41.

Wu, Z. and White, J.D. (1990), “Computer processing of Chinese characters: an overview of two decades research and development”, *Information Processing and Management*, Vol. 26, No.5, pp. 681-692.

Zhou, L. (2001), “Research of segmentation of Chinese texts in Chinese search engine”, *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference*, Tucson, AZ, Vol 4. pp. 2627-2631.

Figure 1: Pootle Interface

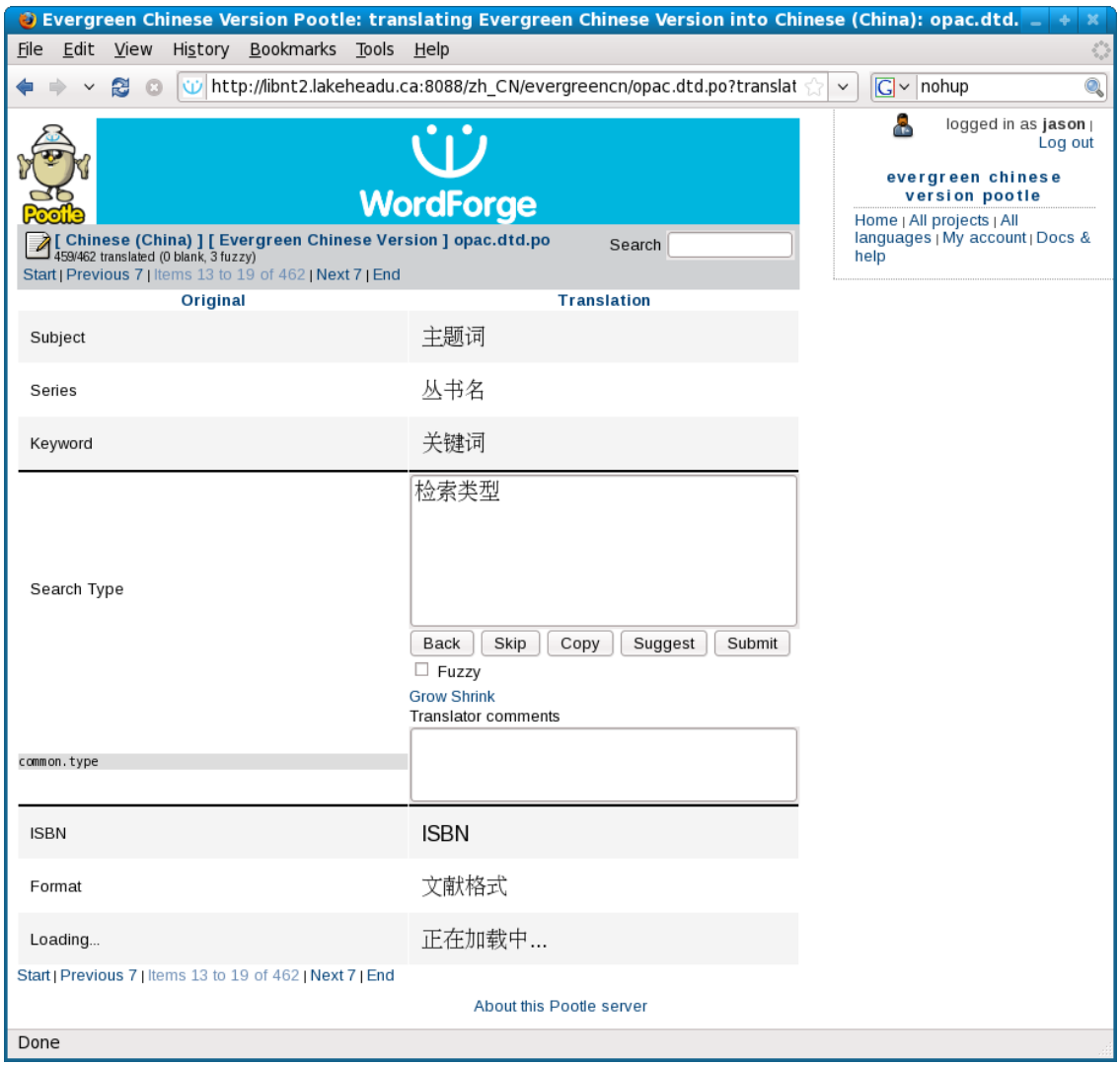

Figure 2: Evergreen result interface