**A new DBMS and flexible POS tagging for EFL learners and researchers**

**Designing a corpus analysis system based on the three-tier model\***

*Takeshi Okada*

Graduate School of International Cultural Studies

Tohoku University

*t-okada@intcul.tohoku.ac.jp*

*Yasunobu Sakamoto*

Faculty of Liberal Arts

Tohoku Gakuin University

*yasube@cs.tohoku-gakuin.ac.jp*

# **Abstract**

This paper reports on our on-going project of designing a new corpus analysis system based on the relational database management system (RDBMS). The introductory section provides an overview of the system to which we try to apply the idea of the so-called three-tier model in order to improve the scalability and performance of the system. The second section presents the present status of the project in which design of the back-end tier (database) and the mid tier (data inputting modules) are almost complete. The third section gives a detailed sketch of the ‘tables’ in the database. By designing a number of tables to reflect the various structural information contained in the corpora, system users would ultimately arrange with ease their target ‘corpus set’, together with POS tags of their own definition. In the last section we give a short summary and address some remaining issues.

# **Introduction: three-tier model and corpus analysis system**

The three-tier model, which is one of the multi-tier architectures, is most widespread in software engineering (Petrou, *et al*. (1999)). In this model with well defined interfaces the layer closest to the system users is often called the front-end (presentation) tier, the layer in the middle is called the mid (logic or application) tier and the layer farthest from the users is called the back-end (data) tier. The greatest advantage of this model is that any of the three tiers can be modified or replaced independently in accordance with the change of user demands or technology. As shown in Figure 1, we have tried to apply the idea of the three–tier model to the integrated corpus analysis system. The Java or the Web-based GUI (graphical user interface) in the front-end tier has been designed to operate mainly on the Internet and users/researchers may ‘interact’ with the system only through the GUI. It should be pointed out that, as observed in Figure 1, the mid-tier consists of more than one separate module, designed to work correlatively with each other in the RDBMS, and that this architecture consequently provides users with a seamless analytic logic for corpus investigations, as well as a data formatting and inputting logic for the database.1 The back-end tier consists of RDBMS through which corpus data is stored, replaced or retrieved by SQL. Since the data is kept independent from the corpus analysis programs, system users with diverse interests can share any data.



Figure 1. The three-tier model and the corpus analysis system

Although a number of fairly large-scaled collaborations on corpus studies have been initiated by world-renowned institutions, many corpus researchers, especially EFL/ESL researchers, to whom we pay special attention in this paper, still have only develop a small-sized corpora to which they give their own investigations. As illustrated in Figure 2, we see how each EFL/ESL researcher tends to compile his/her original learner corpus with a distinct format and with poor compatibility with other researcher purposes.

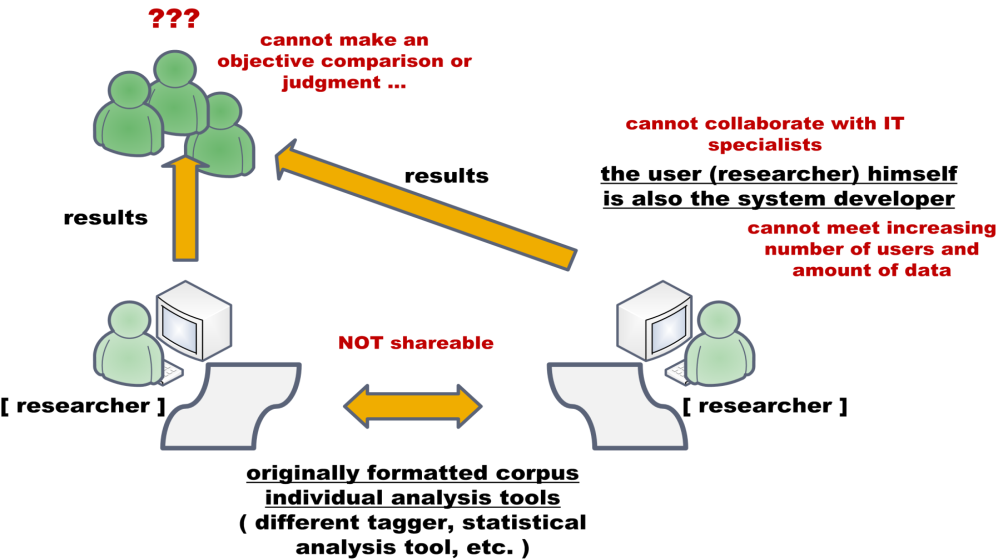


Figure 2. Low scalability of corpus studies

The low uniformity among ‘originally’ compiled corpora most apparently appears on the level of POS tag assignment, since there are a number of different POS tag sets (Atwell (2008)). Even among the POS tag sets belonging to the same family, such as the CLAWS tag sets, there is diversity in terms of their degree of ‘delicacy’. In addition to this diversity of the format, analysis tools (especially statistical ones) tend to be dependent on the purpose of the analysis itself; in a word, there is poor compatibility among analysis tools. This means that each researcher is at the same time a system developer. In turn, this means that the system cannot meet increasing demands since this corpus system development style, by its nature, does not allow many software engineers to be involved as collaborative developers. As a result of the low scalability and incompatibility of the small individual corpus analysis systems, other researchers, i.e. the audience, who receive the results of the analysis cannot make a fair judgement as to reliability, since the results are obtained through individual analysis of individually-compiled corpora. By applying the idea of the three-tier model to the corpus analysis system, we expect that the deficiencies commonly observed in small scale ‘personal’ corpora can be resolved; namely: low scalability, low collaborability and low compatibility.

# **Present status**

This section reports the present status of the project, aimed at designing the back-end tier (the database) and the mid-tier, especially whose data inputting modules, are almost completed. Before we give a closer look at the tables in the database, let us touch briefly on two main roles of the mid-tier: data inputting and statistical analysis. Data input is realised by a (POS) tag assigning process which starts from tokenisation (Schmidt (2008)) and a pre-existing process of analysing existential corpora, such as the BNC.

Since the development of the statistical analysis modules for actual research requires that the database contains a sufficient amount of data, we first completed developing a program which interprets the structural markup of an existential corpus such as the BNC and then extracted the text body and its accompanying attributive information separately. This program is illustrated in Figure 1 as the BNC analyser. Along with it we have already completed the line formatter (tokeniser), whose details will be discussed later. The line formatter, together with the corpus analyser, generates a sufficient amount of input data for the POS tagger, which will be developed in a future phase. In short, our work demonstrates that the database can have a fairy large amount of textual data extracted from the existential corpora and that a considerable amount of data can be obtained through the line-formatting process. To the plain data users can assign POS tags and additional attributive tags which satisfy their specific needs. For example, users are allowed to select a tag set such as CLAWS5 or CLAWS7 and, utilising a program named “POS tag convertor”, they can even convert one tag set which has already been assigned into another sort of tag set. In addition, users can design their own preferred tag set through a program called “POS tag designer”. In this way, if the input data has already been assigned POS tags, they can be interpreted and, if need be, replaced by a different tag set, through the POS tag convertor program. Consequently, the amount of data in the database will increase under a common format to which flexible POS tags and attribution can be assigned.

The system users, especially EFL/ESL researchers or learners, tend to compile relatively small-sized corpus sufficient to satisfy their needs on a specific register or within a particular genre. In order to obtain their own corpus, they have to intake various texts from ‘raw’ resources, such as printed materials, Web documents or PDF documents and convert them into a format suitable for the further compilation and analysis process. For example, printed materials usually have to be OCR-ed and Web documents should be processed carefully since they are usually annotated via some markup language such as HTML or XML. In both cases, tokenisation is a persistent problem, as pointed out in Mikheev (2000), Mikheev (2002), Ratnaparkhi (1996) and Reiley (1986), which cannot be perfectly handled by the software.

The line formatter is developed as an application using Java language. In order to handle the great diversity of raw input data, the actual tokenising process is separated into a series of filters. Since each filter is realised as a Java class, the user can specify the types of filters required and their order of application, thanks to a Java functionality operation called “Dynamic Class Loading”.2 The flexible filtering is based on morphological analysis and is achieved through a scheme of inheritance and polymorphism that is characteristic to an object-oriented language. In other words since the line formatter is well-designed, utilising polymorphism, it can load a queue of filter classes. Each filter functions independently and in accordance with the target input data that the user can specify, as s/he can, their ordering in the queue. 3

In addition to this flexible morphological analysis for tokenisation, this module requires that a new, flexible spellchecking functionality be implemented, as the system should be able to cope with documents created by non-native speakers of English, i.e. EFL/ESL learners.4 An adapted version of spellchecker discussed in Mitton and Okada (2007) and Okada (2009), should be chosen as one of the filters for treating spelling errors in documents created by non-natives.

# **Tables in RDBMS**

Let us give a closer look at the tables in the database. As we postulate two major user groups, corpus researchers, and EFL learners, the system as a whole should be designed to meet the demands associated with users of differing levels. For example, researchers, who simply may be called linguists, would prefer detailed POS tag sets at the sacrifice of mnemonicity in order to analyse the tag sequence by the computer. On the other hand, EFL learners tend to appreciate those tag sets since they intuitively tell them the general usage pattern of a given word or the overall sentence structure in which the word actually occurs; this despite the fact that they are of relatively rough granularity (Granger (1998)). In other words, we have sought to overcome the traditional “trade-off” tagging problem between conciseness and perspicuity by making the system 'friendly' toward users' demands. This we have done via the design of the tables in the database.

Since each document is managed independently in the database, users can select different documents and treat them as a single corpus-set for the purpose of statistical analysis. In each document table, constituents such as paragraphs, sentences, words, and POS tags, as well as diverse attributions, are precisely described through internal links to individual tables, reflecting their relationships.

## **Word list table and POS list table**

We illustrate the tables in our database by providing an example sentence: *“This system is useful for a wide range of researchers.”* Table 1 below is a word list table in which all lexical items (tokens) in the database are listed in random order with a sequential number (named “word key”) in the leftmost column. Table 2 is a POS tag list (in this example, CLAWS5 tag set) where each POS tag is assigned a serial number (named “POS key”). What should be noted here is that the POS tag list table can be modified or replaced with another POS tag set table if the user so wishes. Through the POS tag designer, a user can describe the relationship between the ‘parent’ POS tag and its daughter tags; and a user can even design utterly original POS tag sets of his/her own.



Table 1. Word list table Table 2. POS list table

## **Sentence table**

Each word in the example sentence is given a sequence number together with its word key and POS key as illustrated in Table 3, which describes the sentence numerically.



Table 3. Main sentence table 　　Table 4. Sentence management table

The sentence management table describes the relationship between the lines, i.e. sentences and their constituents, words. Table 4 shows that the sentence given a sequential number 1 begins with a word assigned sequential number 1 and ends with a word with sequential number 11; and the next sentence starts with a word which has sequential number 2 and ends with a word with sequential number 21, and so forth. The first word of the sentence sequence number 1 has a word key 24 which is linked to *this*, together with a POS key 13 (DT0); the second word in the first sentence has a word key 197 (*system*) and a POS key 18 (NN1), and so forth, as observed in Table 3.

## **3.3 Document management table and document attribution table**

As mentioned above, by designing tables which describe the attributive relationships among words and sentences RDBMS can handle them easily. The system can also systematically handle the data if the attribution assigned to each document is successfully expressed in tables, as illustrated here.



Table 5. Document management table Table 6. Document attribution table

From documents tabulated in Table 6, the user can select and specify a group of documents as a corpus-set eligible for the analysis. The system obtains all the necessary information about the documents selected via the relationships represented in Table 5, since the relation is linked to the column entitled ‘Sentence sq(sequence) No’ in Table 4, which, in turn, denotes that, for example, the first sentence of the first documents consists of eleven words together with their word keys and POS tag keys.

## **Lexical relation tables**

Employing the logic of lemma, this proposed methodology can be expanded to cover a variety relationships among lexical items if we allow users to design their favourite lexical relation tables through a program named lexical relation table designer. When we postulate EFL/ESL learners as system users, it is beneficial especially for the basic and intermediate-level learners to have a chance to learn, for example, antonyms, synonyms or verb conjugations in the process of using the corpus analysis system. It is noteworthy here to point out that through tables that can be designed by the users themselves, the delicacy among the lexical relation tables is also finely tuneable to meet users’ demands. In other words the number of columns in the tables, which represent lexical relationships of a given word, must be variable. Therefore, and having found that a single SQL query is insufficient, we introduced a management table for the purpose of retrieving the lexical relationship of a word from variously formatted tables. Tables 7, 8, 9 and 10 below are examples of lexical relation tables that are linked to each other. Table 7 is the detail of the main sentence table, presented in 3.2, but includes two additional fields which show the lexical relationship of each word. Note that the parenthetical items in the second and third columns of Table 7 are for explanatory convenience; they are actually all numericals. In addition, the labels of each field in Tables 9 and 10 are not fixed POS tag names; they can be defined by the users. Thus the system can retrieve all words having a lexical relationship with a given word from user-defined tables.



Table 7. Main sentence table (detail)

Word sq No: record number for each word

Rec No(A ): record number of Table 8 standing for the table in which a lexical relationship of the word is defined

Rec No(B): record number in a table indicated by Rec No(A) including all relationships of the word



Table 8. Lexical relation management table



Table 9. Lexical relation tables (noun table)



Table 10. Lexical relation table (verb table)

To better illustrate, let us take a word *love* for example. The fourth column of Table 7 shows that *love* has a link to two distinctive lexical tables, i.e. 1 and 2, which correspond to the category noun and verb, respectively, as observed in Table 8. Table 8 also shows that the verb table may have two potential items, i.e. singular and plural, and that the verb table can have seven items which correspond to conjugational forms. Therefore, the relationship among these tables indicates that the noun *love* has two tokens with POS tag NN1 and NN2 respectively, and that the verb *love* can be potentially manifested in seven conjugational forms with corresponding POS tags.5 The most important thing to be mentioned here is that each of the tables can be designed by the users themselves. This means that each user can control the relationship among words, as well as the delicacy of the POS tag set.

# **Conclusions and further issues**

The flexibility of the tables described so far reflect the attempt to satisfy the preferences and needs of system users, who we assume possess diverse interests. Each table, which is systematically linked together in RDBMS, would function as a robust annotation system capable of satisfying users’ demands. Moreover, due to the inter-relationality of the tables, users can even specify their desired attribution header for any text. This makes cross-register/genre-statistical investigations feasible, and the results of analyses thereby may contribute to the work of other system users across a number of diverse contexts. Since this paper reports the present status of our on-going project, there remain several issues to be discussed regarding the next phase. For example, in designing RDBMS tables for the corpus study, the effect of RDBMS normalisation is one of the crucial factors we should keep in mind; further, we have to conduct a series of performance tests of RDBMS in order to check whether our table design is useful and capable of accommodating the entire corpus analysis system. SQL, which we use to retrieve data from the corpus, does not have any characteristics specifically connected to the corpus data. On the other hand, as the corpus is an electronic compilation of naturally-occurring language data, one can anticipate that it would inevitably contain unique characteristics peculiar to natural language, where the constituents, such as words, phrases, clauses or sentences, tend to have diverse, cross-cutting connections with each other. Moreover, it would be preferable for these constituents to be annotated individually and, as we saw in Table 6, the entire documents themselves require attribution. A systematic way to render a database structure suitable for general-purpose querying via SQL would be to a process called “normalisation”, which is believed to reduce data redundancy and guarantee data integrity in RDBMS. However, in the past it has been common for a logical design of RDBMS and actual implementation of the corpus data to conflict. In other words, to describe in the RDBMS all the diverse information the corpus should contain is a very difficult, complex task. Accordingly, since a *completely* normalised RDBMS has to possess highly complex relationships amongst tables and demonstrate that it can support multiple queries in retrieving target information from the RDBMS, we must undertake a number of performance tests.

Another remaining problem is designing tables which represent attributive information given to individual sentences and words in the documents. It is indispensable for current corpus studies to attend to the attributive information: page layout, such as itemisation, and orthographical decoration, such as word emphasis in the form of bold face or italic face. If our RDBMS successfully represent such attributions in the tables, users can easily separate, for example, the body and the headlines of newspaper documents. As we briefly mentioned in section 2, Web documents, annotated (marked up) by XML, can be input sources for the database. Thus, if we can design a series of attribution tables which take care of markup tags in XML, the above-mentioned differentiation among heterogeneous constituents would be possible.

Lastly, since we postulate two major types of system users, i.e. EFL/ESL learners and researchers, it is definitely important to perform thorough surveillance studies in order to determine such users’ actual demands.

# **Notes**

\* This project is partly funded by a Grant-in-Aid for Scientific Research (C) [#19530773l] from the Independent Administrative Legal Entity Japan Society for the Promotion of Science.

1. For further discussion of data formatting including tokenisation, see Okada (2009).
2. See Ramirez (2000) for further details.
3. It might be a more realistic solution to allow the user to make a decision, through GUI, regarding differentiation between sentence-final periods and periods accompanying abbreviations.
4. For a detailed discussion on spelling errors made by Japanese writers of English, see Okada (2003) and Okada (2005).
5. These POS tags are flexibly designed by users using the POS tag designer mentioned in Section 2.

# **References**

Atwell, E. (2008). “Development of tag sets for part-of-speech tagging”. In A. Lüdeling and Kytö, M. (ed.) *Corpus Linguistics: An International Handbook (Volume 1)*. Berlin: Walter de Gruyter, 501-527.

Granger, S. (1998). “The computer learner corpus: a versatile new source of data for SLA research”. In S. Granger (ed.) *Learner English on Computer*. London: Longman, 3-18.

Mitton, R. and T. Okada (2007). “The adaptation of an English spellchecker for Japanese writers”. In *Proceedings of the Symposium on Second Language Writing: Second Language Writing in the Pacific Rim,* 6, 15.

Mikheev, A. (2000). “Tagging sentence boundaries”, In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics,* 264-271.

Mikheev, A. (2002). “Periods, capitalized words, etc”. *Computational Linguistics,* 28, 289-318.

Okada, T. (2003). “Spelling errors made by Japanese EFL writers: a corpus-based approach”.paper presented at Colchester Second Language Acquisition Workshop 2003, University of Essex, Colchester, UK.

Okada, T. (2005). “Spelling errors made by Japanese EFL writers: with reference to errors occurring at the word-initial and the word-final position”. In V. Cook and Bassetti, B. (ed.) *Second Language Writing Systems*. Clevedon: Multilingual Matters, 164-183.

Okada, T. (2009). “An adaptation of English spellchecker in new corpus analysis system based on the three-tier model”, *Journal of the Graduate School of International Cultural Studies*, 17 (forthcoming).

Petrou, C., S. Hadjiefthymiades and D. Matrakos (1999). “An XML-based, 3-tier scheme for integrating heterogeneous information sources to the WWW”, In *Proceedings of Tenth International Workshop on Database and Expert Systems Applications: 1999*, 706-710.

Ratnaparkhi, A. (1996). “A maximum entropy model for part-of-speech tagging”, In *Proceedings of the Conference on Empirical Methods in Natural Language Processing,* 133-142.

Reiley, M. D. (1989). “Some applications of tree-based modelling to speech and language indexing”, In *Proceedings of the DARPA Speech and Natural Language Workshop,* 339-352.

Ramirez, A. O. (2000) “Three-tier architecture”, *Linux Journal (July 1st, 2000)*

Available at: http://www.linuxjournal.com/article/3508 (accessed: 2 Sept 2009)

Schmid, H. (2008). “Tokenizing and part-of-speech tagging”, In A. Lüdeling and Kytö, M. (ed.) *Corpus Linguistics: An International Handbook (Volume 1)*. Berlin: Walter de Gruyter, 527-551.