A Software Toolkit for Sharing and Accessing Corpora Over the Internet

Saturnino Luz

Natural Interactive Systems Laboratory
University of Southern Denmark
Forskerparken 10
5000 Odense, Denmark
luzs@acm.org

Abstract

This paper describes the Translational English Corpus (TEC) and the software tools developed in order to enable the use of the corpus remotely, over the internet. The model underlying these tools is based on an extensible client-server architecture implemented in Java. We discuss the data and processing constraints which motivated the TEC architecture design and its impact on the efficiency and scalability of the system. We also suggest that the kind of distributed processing model adopted in TEC could play a role in fostering the availability of corpus linguistic resources to the research community.

1. Background

Corpus linguistics has gained increasing importance in both theoretical and computational linguistics. Witness the considerable volume of corpora and software tools for corpus processing currently on offer from linguistic data associations in Europe and the US.

Very large corpora such as the Bank of English have been used for some time in the area of lexicography for analysing collocation patterns (Clear, 1993), normally in conjunction with computational and statistical methods[1] . More recently, there has been a tendency towards the application of similar corpus analysis techniques to smaller corpora in order to study language use in more specialised domains.

Whatever the domain, corpus research requires efficient access to large volumes of linguistic data, imposing strict data and processing constraints to the supporting software. Traditionally, corpus undertakings have focused on building large, centralised repositories of text accessed via corpus-specific tools. Both data and tools are normally distributed to the public through physical media such as CDROM's, or made available by granting users access to machines where the corpus is physically stored. In this paper we argue that useful and efficient access to considerable amounts of (written) linguistic data does not necessarily entail accessing vast, centralised repositories. We present a software architecture for corpus maintenance and use set on a highly distributed environment which aims at allowing uniform and seamless access of language resources to large user groups. The architecture is illustrated by the implementation of TEC (the Translational English Corpus) software toolkit.

TEC is a corpus of contemporary translational English: it consists of written texts translated into English from a variety of source languages. The corpus is designed to allow researchers to study the distinctive nature of translated text. About 80% of TEC consists of translated fiction. Most of it is copyrighted, which rules out its distribution through physical medium as well as direct access to the text. The scenario to be discussed in this paper assumes that a number of medium-sized corpora such as TEC, located at different sites, possibly maintained by different organisations, is to be made available for analysis to the research community in a way that satisfies copyright constraints.

The underlying processing model of the TEC toolkit comprises an extensible client-server architecture which supports the several corpus analysis functions across a variety of platforms. The basic functionality for corpus indexing and access is handled at the server side while the client handles not integral texts directly but the concordances generated by the server, thus respecting copyright restrictions. The client itself is implemented in a modular architecture which enables new functionality to be incorporated seamlessly.

2. The Translational English Corpus

The basic structure of TEC was first proposed in (Baker, 1995) but the design principles have since been elaborated in greater detail in (Laviosa-Braithwaite, 1996; Laviosa, 1997; Laviosa, 1999; Baker, in press). The corpus is currently held at the Department of Language Engineering, University of Manchester Institute of Science and Technology (UMIST).

The corpus consists of unabridged, published translations into English from European and non-European source languages. Four text categories are currently represented in TEC: biography, fiction, newspapers and inflight magazines. The current size of the corpus nears 6 million tokens. New data is being added to the corpus on a regular basis, so this figure and the size of the corpus will continue to grow in the next few years.

TEC is designed to allow researchers to study the distinctive nature of translated text (as opposed to original text production). The starting point in this type of research is that translation is a distinct type of communicative event, shaped by its own goals, pressures and context of production. For example, a translation is generally constrained by

[1] See, for instance, (Mason, 1997) for a description of a toolkit used in the analysis of the *Bank of English*, a corpus of over 400 million words.

the presence of a fully articulated text in another language; this is not true of original text production.

Moreover, translators are likely to respond, consciously or subconsciously, to the perceived social status of the text they are producing (Baker, 1995; Baker, in press). They know that translations are not normally received in the same way as original texts in most contemporary societies, though this perception of course differs from one social and historical context to another. Their awareness of this special context of reception may explain a feature which is associated in the literature with translated text, namely that it tends to conform to, and even exaggerate, typical patterns of the target language.

Constraints of this type then must leave traces in the language that translators produce, the analysis of which should give us some insight into the processes that shape translational output. This issue is not of concern only to translation scholars: all linguists must sooner or later come to terms with the fact that much of the language we all consume on a regular basis now is translational in essence, and it is imperative therefore for anyone with an interest in language to study the language of translation in its own terms.

3. The TEC software tools

The software components of TEC were designed to enable multiple users to access a common corpus, retrieve concordance sets and perform operations on these sets such as sorting, collocation, tagging, and statistical analysis. The architecture chosen to implement this functionality follows the client-server model depicted in Figure 1. The basic functionality for corpus indexing and access is handled at the server side while the client handles not the original texts directly but the concordances generated by the server.

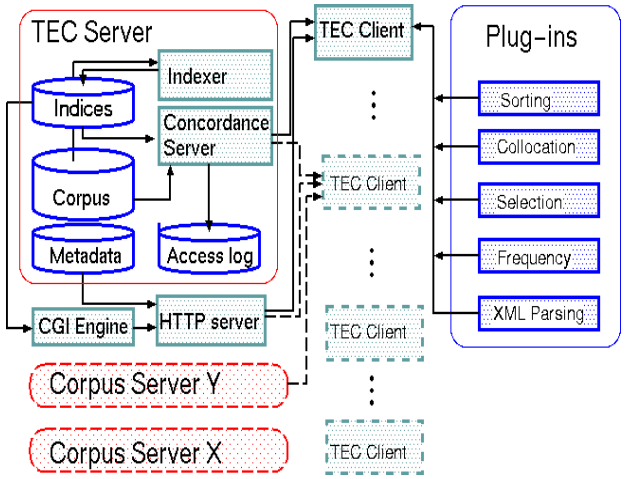

Figure 1: TEC Architecture

The client itself is implemented in a modular ("plug-in") architecture which enables new functionality to be incorporated seamlessly. The core function of the client is to manage user requests and the communication with the server. Once the data has arrived at the client end the appropriate plug-ins take over the processing of the request.

The TEC client may also communicate with a standard HTTP server for requests which involve non-indexed text *about* the corpus. This kind of text will possibly contain mark-up tags which need to be parsed appropriately. The reason for splitting the server function between a specialised concordance retrieval and a generic content server is twofold. Since media access is clearly a bottleneck in corpus processing, the concordance server can perform more efficiently if it is dedicated to retrieving concordance data. Furthermore, TEC is largely a corpus of copyrighted material, therefore unrestricted access to the texts cannot be allowed.

These and other data and processing constraints are spelled out in the next sections.

3.1. Constraints on data storage and indexing

As mention above, 80% of TEC consists of translated fiction (including bibliography). All this material is subject to copyright restrictions. In order to be able to make the corpus accessible as a resource for researchers and translators while abiding by the copyright law it was necessary to protect the original texts. This was done by allowing only indirect access to the texts, via the concordance server and browsers. The system architecture described above satisfies this constraint nicely, since the clients only receive lists of words drawn from a variety of texts and their immediate contexts.

Given the size of the corpus, efficient access to concordance data necessarily implies indexing. The TEC indices are pre-compiled and updated every time text is added to the corpus. The server accesses the indices directly, upon request from a client. The current implementation of the server stores large chunks of index tables in memory to minimise disk access. However, we intend to replace this lookup strategy by one based on hash tables tied to disk which will enable us to update the indices without having to restart the server as well as spare more working memory for multi-threaded processing. Hash table lookup can be made quite efficient and many compression techniques exist which can be used to reduce index space requirements (usually quite high) and minimise the physical media access bottleneck.

```
<omit desc="caption">
  Chancellor Kohl . . . through all the embarassing twists, still the
  dominant player
</omit>
<p>
<omit desc="title">KING MAKING</omit>
<p>
<omit desc="synopsis">
  Will Chancellor Kohl's vision of democracy prevail in united
  Germany's first presidential election? Robert Leicht argues that the
  liberals' time has come
</omit>
<p> Die Zeit
<p> THE GERMAN head of state has no power. But a
presidential election tells us much about power in this land - and the
health of its political culture. On Monday the first president of a
reunited Gemany will be elected - but the heavily symbolic aspect of
this no longer plays a role in the outcome. For the second time in the
history
[...]
```

Table 1: Excerpt of a TEC text file

In addition to the original text, for each TEC file we record extra-linguistic attributes which concern the translator, the translation itself, the translation process, the author, and the source text. These attributes are recorded in a separate header file. There are two types of header. One is for

single volumes such as novels and biographies, the other is for collected works, such as collections of short stories by different authors and/or translators or a collection of newspaper articles published in the same newspaper issue.

A TEC user may wish to compare lexical density, type/token ratio, sentence length, collocational patterning, or the use or frequency of specific lexical items in:

- texts translated by male versus female translators,
- texts translated from a particular source language,
- sub-corpora of texts translated from different source languages,
- translations produced by a particular translator, etc

The meta-information carried by the header is the primary source for building index tables to support this kind of specialised query.

Both text and header files are annotated. Header files are heavily annotated in order to provide fine granularity access to extra-textual data (see table 2). Text files are lightly annotated (see table 1). The primary goal of annotating TEC text files is to preserve the integrity of the translated text. Non-translated material such as an editor's note, a preface written in the target language, etc are marked-up and ignored by the indexing program so that they do not produce concordances or entries in the frequency list.

```
<tecHeader>
  <title>
    <filename>fn000011.txt</filename>
    <subcorpus>fiction</subcorpus>
    <collection>Lúcio's Confession</collection>
    <editor></editor>
  </title>
  <translator>
    <name>Margaret Jull Costa</name>
    <gender>female</gender>
    <sexualOrientation>heterosexual</sexualOrientation>
    <Nationality>British</Nationality>
    <employment>Translator</employment>
    <status>full-time; free-lancer</status>
  </translator>
  <translation>
    <mode>written to be read</mode>
    <extent>27770</extent>
    <publisher>Dedalus<publisher>
    <pubPlace>UK</pubPlace>
    <date>1993</date>
    <copyright>Dedalus</copyright>
  </translation>
  <translationProcess>
    <direction>into mother tongue</direction>
    <mode>written from written source text</mode>
    <type>full</type>
  </translationProcess>
  <author>
    <name>Mario de Sá-Carneiro</name>
    <gender>male</gender>
    <Nationality>Portuguese</Nationality>
  </author>
  <sourceText>
    <language>Portuguese</language>
    <mode>written to be read<mode>
    <status>original<status>
    <pubPlace>Portugal</pubPlace>
    <date>1913</date>
    <comments></comments>
  </sourceText>
</tecHeader>
```

Table 2: A TEC header

TEC annotations are encoded using the Extensible Mark-up Language, XML (Bray et al., 1998). Table 1 shows an excerpt of a newspaper article which forms part of TEC. Table 2 shows the header of a translated fiction text in the same corpus.

3.2. The corpus processing model

The processing model employed in TEC comprises two distinct phases: index compilation and on-line concordance browsing.

The index compilation phase is performed off-line by the maintainer of the corpus and precedes the use of the system by the final user. Indexing is done by two main modules: a *frequency list generator* and an *index builder*. The frequency list program performs the task normally performed by the lexer (or tokeniser) of a compiler. It uses regular expression matching to break the text down into tokens, counting each occurrence and building a frequency table in the process. The indexer then reads in this frequency table and generates an inverted index, storing the exact coordinates of each token.

On-line corpus usage at present involves both "static" and "dynamic" components. Static browsing refers to access of meta-data or pre-computed statistics and is done via standard HTTP. Dynamic browsing refers to retrieval of indexed data by the server and post-processing by the client.

The dynamic component is a mobile Java client which manages user input by contacting the server (through TCP/IP sockets), placing a request for a corpus search and displaying the output onto the user's screen. The syntax for user queries currently includes case sensitive and insensitive search as well as "wild-cards".

The static component is available through the world wide web and can be accessed with most ordinary web browsers. Its back-end is a CGI script which processes precompiled frequency lists and header files.

A screen shot of the actual browser displaying a concordance and the header information for one of the lines is shown in Figure 2. The concordance browser is available as a stand-alone application and as a Java *applet* running on top of a Java-enabled web browser in order to enhance its portability across platforms.

Attached to the client are several smaller software components (the plug-ins shown in Figure 1) which deal with further processing of the data sent by the server on the local (client) machine. The system currently incorporates the ability to sort concordance lists, activate and de-activate (non-validating) XML parsing of text files and headers, extract wider contexts, and save concordances onto the local disk for further use (option available only in the stand-alone version, which is freely available at the TEC web site).

A typical use case is the following:

- the user requests a concordance for a particular word,
- the server starts to transfer to the client a (potentially very large) list of words and their associated context,
- the client starts to display the concordances on the users screen as soon as they arrive, keeping a separate thread of processing running in order to handle the remaining data,

- the user performs actions on the concordance (or subsets of it) such as: sorting it by the n^{th} word to the left of right of the keyword (context horizon), inspecting extra-linguistic information associated with a particular occurrence of a word, inspecting larger contexts to the left and right of the keyword etc.

Given this scenario, it is reasonable to assume that the browser interface could be improved by allowing each plug-in to spawn its own processing threads as well as run them independently from the network. This is one of the improvements planned for future releases of the software.

4. Distributed processing

The architecture described above can be used in support of two related processing models: a model of *distributed processing of a corpus* and also one of *processing of distributed corpora*.

The former is discussed in sections 3.1. and 3.2., above. It is represented in Figure 1 by the the boxes labelled "TEC Server" along with the several TEC clients, plug-ins and connectors. What makes this a *distributed* architecture is the fact that the computational cost of retrieving and handling concordances is shared among different processes, running on multiple machines, possibly on different hardware and software platforms while preserving the uniformity of the user interface (Lamport and Lynch, 1990). If one removes the extra instances of the client and interprets the connectors as information flow on the same processor, one gets a standard non-distributed corpus processing architecture, such as the one assumed in (Mason, 1997).

We believe that the architecture employed in TEC provides a more efficient and scalable processing model. In this model, a more powerful machine (the server) can be allocated the (memory- and disk-intensive) tasks of searching for and retrieving concordances while clients can take care of the (potentially CPU-intensive) job of displaying, grouping and manipulating chunks of text handed over to them by the server.

A drawback of the current implementation of the TEC processing model is that it introduces an additional bottleneck, namely, the speed of the network connection between client and server. However, with the rate at which bandwidth has increased in the internet over the past years, this limitation tends to be naturally minimised by improvements on the underlying structure of the network. Furthermore, the compression techniques currently used to mitigate delays due to disk access to large files[2] can also be used to reduce the amount of data sent over the network. All that is necessary to add to the client is a decompression plug-in. Further performance gains can be achieved via client-side disk caching and other techniques used in standard WWW browsers.

Another attractive aspect of the TEC architecture is that it scales up at the server side as well as it does at the client side. Therefore the architecture can evolve into one where the location of corpora (and indices) themselves is distributed. Several corpora could be kept at different sites and maintained by different organisations (represented by the boxes labelled "Server X" and "Server Y" in Figure1). Each client will be able to choose a set of corpora to query, and the query will then be broadcast to the chosen servers. Each server deals with its own query — which introduces a rudimentary form of *parallelism* to the model[3] — and sends the information back to the client which merges and present the data to the user.

Heterogeneous implementations in support of this model of processing of distributed corpora can be accommodated with existing technology – e.g. by adding a request brokering layer such as CORBA (*Common Object Request Broker Architecture*) between servers and clients.

5. Conclusions and further work

The TEC server has been implemented in Java and currently runs robustly on a rather modest hardware setting (a 166MHz Linux PC with 128Mb of RAM). The CGI layer along with the indexer and other maintenance programs have been implemented in Perl. The client has been fully implemented in Java in order to enhance its portability to other platforms. Currently both an applet and a stand-alone version of the client are supported. They applet can be accessed via

http://ubatuba.ccl.umist.ac.uk/tec/

The TEC browser applet should run on any Java-enabled browser. The stand-alone version is also found at the URL above for download and requires JRE (Java Runtime Environent) to be installed. The source code and documentation for both client and server are freely available for use, modification and redistribution under the Gnu Public License (GPL).

TEC has considerable potential for stimulating a variety of studies into the language of translation. We presented above a few suggestions about the possible lines of research that can be fruitfully pursued. Some of these suggestions are not feasible using the software currently available on the site, but the software is being enhanced to allow these and other analyses in the very near future. It must be pointed out, however, that in spite of the present limitations of the TEC software, some studies have already been undertaken which used the corpus and its web-based tools. These include parts of two PhD theses: (Laviosa-Braithwaite, 1996) and (Kenny, 1999), and a study of the distinctive stylistic features of individual translators (Baker, forthcoming).

In addition to the performance improvements discussed above, functionality enhancements such as the ability to select subsets of the corpus and analyse collocation patterns on selected concordance data appear high in our list of features to be incorpoated to the next release of the TEC tools.

Ideally, to make the best use of TEC, the researcher should also have access to a general monolingual corpus of English, such as the British National Corpus. The only way to establish whether a particular pattern is specific to

[2]Compression rates of over 70% can be achieved efficiently using the Burrows-Wheeler block-sorting text compression algorithm, and Huffman coding.

[3]Not to be confused with "parallel corpora", which refers to collections original texts aligned with their corresponding translations in one or more languages.

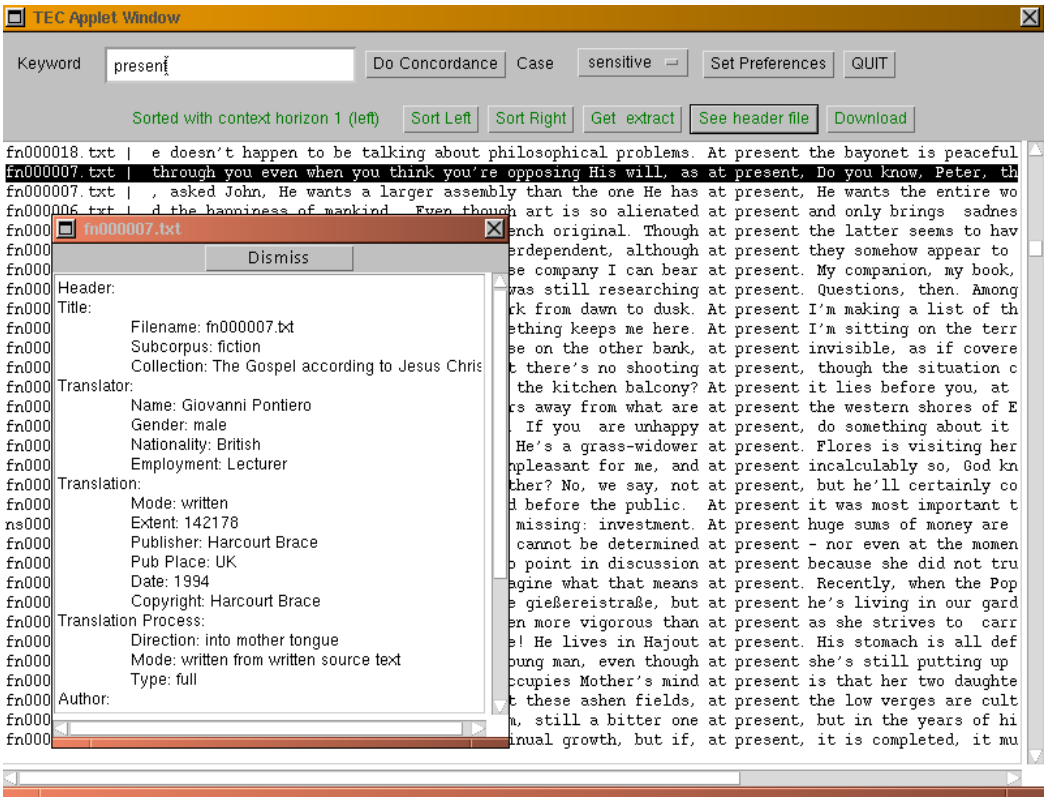

Figure 2: The TEC Corpus Browser

translated text is to check it against the patterning of non-translated text in the same language. Perhaps in a not so distant future this kind of research may be facilitated by a "world wide web of corpora" in which browsing through different corpora sitting in in different parts of the world will be just a click away.

6. References

Baker, Mona, 1995. Corpora in translation studies: An overview and some suggestions for future research. *Target*, 7(2):223–243.

Baker, Mona, forthcoming. Do literary translators have style. *Style*.

Baker, Mona, in press. The role of corpora in investigating the linguistic behaviour of professional translators. *International Journal of Corpus Linguistics*.

Bray, Tim, Jean Paoli, and C. M. Sperberg-McQueen, 1998. Extensible markup language (XML) 1.0. Technical Report REC-xml-19980210, W3C. http://www.w3.org/TR/REC-xml.

Clear, J., 1993. From firth principles: Computational tools for the study of collocation. In M. Baker, G. Francis, and E. Tognini-Bonelli (eds.), *Text and Technology: In honour of John Sinclair*. Amsterdam: John Benjamins, pages 271–292.

Kenny, Dorothy, 1999. *Norms and Creativity: Lexis in Translated Text*. Ph.D. thesis, Centre for Translation Studies, Department of Language Engineering, UMIST.

Lamport, L. and N. Lynch, 1990. Distributed computing: Models and methods. In *Handbook of theoretical computer science*, volume Vol.B Formal models and semantics, chapter 18. The MIT Press: Cambridge, MA, pages 1159–1199.

Laviosa, Sara, 1997. How comparable can comparable corpora be? *Target*, 9(2):289–319.

Laviosa, Sara, 1999. The english comparable corpus: A resource and a methodology. In L. Bowker, M. Cronin, D. Kenny, and J. Pearson (eds.), *Unity in Diversity: Current Trends in Translation Studies*. Manchester: St. Jerome Publishing.

Laviosa-Braithwaite, Sara, 1996. *The English Comparable Corpus (ECC): A Resource and a Methodology for the Empirical Study of Translation*. Ph.D. thesis, UMIST, Manchester, UK.

Mason, Oliver, 1997. CUE – a software system for corpus analysis. In *Proceedings of the 2nd. TELRI Seminar*. Kaunas.