Processing corpora with *Corpus Presenter*

Raymond Hickey
English Linguistics, Essen University

Abstract. The present article offers a description of a new software package – *Corpus Presenter* – which the author has written and which is intended to render the processing of corpora as direct and simple as possible, while offering a range of options which would make it attractive to linguists involved in either the compilation and/or the processing of corpora. Particular emphasis has been laid on the retrieval of information from corpora, especially for linguistic purposes. Provision has been made for the retrieval of syntactic information with frame searches. The processing of lexical information is facilitated by the availability of a number of database modules within the programme suite. The *Corpus Presenter* package also allows tagging of corpora, in an automatic, semi-automatic or manual mode, so that it can be useful to those linguists compiling corpora in which grammatical information is to be incorporated in advance of distribution. The means of linking existing corpora with the *Corpus Presenter* suite is described at the end of the article.

1 Introduction

The intention of the present article is to describe a new software package, which is available to the community of linguists involved in corpus processing and to show by some illustrations just how it can be put to good use in day-to-day work on text corpora. The package is called *Corpus Presenter* and consists of some 20 programmes, which fulfil various tasks in the field of corpus processing (more on this below). In terms of the software available from the present author over the past decade or so, the present suite can be seen as the successor to the package *Lexa* (Hickey 1993a; see Hickey 1993b for a brief description). The latter was initially produced under the older operating system MS-DOS. The currently available version 7.0 (enclosed on the *ICAME Collection of English Language Corpora*, 2nd edition, University of Bergen, Norway) is a considerable improvement on earlier versions in terms of capacity and flexibility, but as there is no

gainsaying the obvious advantages of a graphical, 32-bit operating system like Microsoft *Windows,* the present author decided to design corpus processing software for this latter system. It quickly became obvious that it would not be sufficient to simply revamp the older package and offer it under *Windows*. Instead the author returned to the drawing board and designed the entire suite of programmes afresh, utilising to a maximum the possibilities of the newer operating system. The result is a set of programmes which in their functionality contain more or less all the options of the older *Lexa* package, but very many more as well and with a 'look and feel' which is commensurate with what users have rightly come to expect of software running under Microsoft *Windows*.

1.1 Programme description

The *Corpus Presenter* suite consists of programmes which are dedicated to various related functions. They can interact with each other in several ways, eg by using the same data stored to disk or clipboard. An example of this is the *Corpus Presenter Table Editor,* which allows users to edit the results of retrieval tasks which have been stored in table form to disk from within *Corpus Presenter.* Another example of this interlocking of programmes can be seen with *Corpus Presenter Create* which facilitates the linking of one's own corpus with *Corpus Presenter* by constructing the data set file needed to control the display and manipulation of a corpus internally in the latter programme. There follows a list of the items of the suite, grouped according to function with a brief description of each programme.[1]

Group 1: *Viewing/processing corpora and related data*

1) Corpus Presenter (main programme)
2) Corpus Presenter Create
3) Corpus Presenter Slide
4) Corpus Presenter Quick Note
5) Corpus Presenter Quick Viewer
6) Corpus Presenter Dictionary Viewer
7) Corpus Presenter Table Editor

Group 2: *Managing data on one's computer*

8) Corpus Presenter Launcher
9) Corpus Presenter File Manager
10) Corpus Presenter Quick Backup
11) Corpus Presenter Find Text
12) Corpus Presenter Catalogue

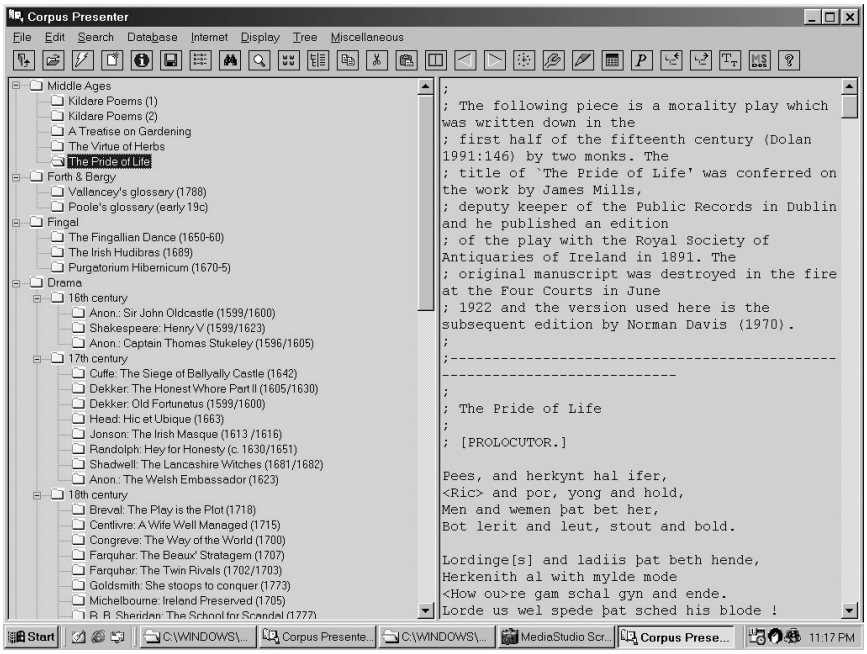
13) Corpus Presenter Diary
14) Corpus Presenter Direct Viewer

Group 3: *Dealing with databases*
15) Corpus Presenter Database Manager
16) Corpus Presenter Make Database
17) Corpus Presenter Quick Database
18) Corpus Presenter Report Database

Group 4: *Processing texts*
19) Corpus Presenter Edit
20) Corpus Presenter Word Processor

1) *Corpus Presenter*
The main programme of the current suite is called *Corpus Presenter.* With it one can carry out all the processing tasks with a corpus of one's own or one to which one has access. If one does not have a corpus one can still load a text directly and carry out retrieval operations. To create the file necessary to process a corpus with *Corpus Presenter,* one uses the programme *Corpus Presenter Create* (see the next programme description).

Within the main programme the structure of a corpus is visible from the tree on the left-hand side of the screen. By moving in this tree, one can view the various files which are associated with the nodes of the tree (each node contains a descriptive reference to a particular file). For a corpus consisting of text files, these texts are displayed in a window on the right-hand side of the screen.



An essential feature of *Corpus Presenter* is its ability to cope with files of different medium types. It can present text files, images (maps, pictures, etc), databases (eg bibliographies) and sound files (eg language samples). These additional types have not perhaps been envisaged by linguists so far, but certainly the option of including images – say facsimile pictures – into a corpus might be appealing in future. Equally for contemporary corpora, the option of including sound files would be enriching in a respect which is central to language studies. For instance, one could imagine offering a version of the London-Lund corpus with the sound files from which the printed transcriptions were derived. The test corpus shipped with *Corpus Presenter* has a number of sound files to illustrate how this option works.

The programme recognizes multi-media files automatically and presents them appropriately. Image files are normally in the *Windows* Bitmap (`.BMP`) or the JPEG Image File (`.JPG`) formats (though other common formats such `GIF`, `TIF`, `WMF` or `PCX` are also accepted). Databases should be in *dBASE* (`.DBF`)

format and audio files in the *Windows* Wave (`.WAV`) format (technical note: these can be compressed into the MP3 format to save disk space and still be accepted by *Corpus Presenter*).

For text files, two special types are automatically recognized: RTF and HTM(L) files. A HTM(L) file is in the Hypertext Markup Language format and can be read and edited by most advanced word processors and by internet software. An RTF file is in the Rich Text Format and can equally be read without difficulty by the majority of commercially available word processors. In addition, a corpus may contain plain text files. Indeed, this is frequently the default case: very often no formatting specific to any word processor is included in a corpus to ensure that the texts can be read on any computer system. *Corpus Presenter* can of course handle plain texts equally well. Such texts can, if necessary, be edited using the supplied text editor *Corpus Presenter Edit,* which can process ASCII and RTF files. The supplied word processor *Corpus Presenter Word Processor* can additionally deal with HTM(L) files. Databases can be edited by several programmes, including two dedicated database managers (see programme descriptions below).

Apart from presentation, the main operation which users will probably be interested in is searching texts. There is a particularly flexible search algorithm built into *Corpus Presenter.* This is discussed in section 2 *Using Corpus Presenter* below.

2) *Corpus Presenter Create*
In order for *Corpus Presenter* to process any set of texts it must have access to a small file called a data set file, which contains a list of the files of a corpus, labels for the nodes in a tree with which the texts are associated, and information on the level in a tree structure at which a label is to appear. In addition, a data set file contains information pertaining to the general appearance of the corpus when it is displayed on screen. Such a data set file can be designed interactively with the current utility.

3) *Corpus Presenter Slide*
One may find that one needs to present the data from a corpus in public, such as at a conference or for a lecture or in a classroom. The present utility will group any set of files into a list which one can page through like slides on a projector (from one to the next, without interruption, on a clear screen). Sample data to illustrate the functioning of this programme is supplied with the suite.

4) *Corpus Presenter Quick Note*
The purpose of the present utility is to allow one to maintain texts with an internal hierarchical structure (such as sections of a corpus) and have these displayed by means of a tree through which one can navigate easily. For this to work, input texts must have Table of Contents markers embedded in them (This can be realised interactively or with the text editor *Corpus Presenter Edit.*).

5) *Corpus Presenter Quick Viewer*
This programme will display any text with an internal structure in the form of a labelled tree which reflects the organisation of sections of the text. When designing a corpus, one could employ the present programme to illustrate the structure of the corpus without having to use *Corpus Presenter* for this task.

6) *Corpus Presenter Dictionary Viewer*
The dictionary viewer is a programme, which will display definitions contained in a single file which is loaded from disk. This file consists of headings and entries, and users can easily create dictionaries of their own with customised data, eg corpus material, glossaries or the like.

7) *Corpus Presenter Table Editor*
Tables are structures where data is presented in the form of rows and columns. This is the primary form in which to save retrieval returns within *Corpus Presenter.* Such finds can be loaded from disk into the present programme and further processed. One can create new tables, copy data to and fro and interface with databases if one wishes.

8) *Corpus Presenter Launcher*
The aim of the present utility is to offer users a programme from which they can then launch any of the elements of the *Corpus Presenter* software suite. It represents the best way to get acquainted with the *Corpus Presenter* suite as it activates any programme at the click of a button and offers brief descriptive texts which indicate what the different items of software can be used for.

9) *Corpus Presenter File Manager*
A file manager is necessary for all the house-keeping tasks which one has to carry out on a computer. This utility has many special features such as incremental backup which is useful when dealing with large amounts of text, such as

in a corpus, which may be variously modified during work sessions and hence in need of backup to a permanent separate medium such as high-capacity disks.

10) *Corpus Presenter Quick Backup*
This programme is similar to the file manager but slightly different in its organisation. Essentially, it allows one to make tables of files altered on a computer and then copy selected items to a backup medium.

11) *Corpus Presenter Find Text*
Normally when compiling a corpus one is dealing with several texts, and it may often be necessary to search for strings across the entire group or even through a complete drive. The present programme will perform this task. A range of options make it a flexible tool for text retrieval.

12) *Corpus Presenter Catalogue*
When processing data, it is useful to group sets of data into larger units. This makes it easier to grasp the organisation on one's computer. The current utility allows one to create catalogues of data sets which can then be viewed with some other programme, such as *Corpus Presenter Quick Viewer*. Sample data, illustrating the functions of this programme, is supplied.

13) *Corpus Presenter Diary*
For good measure this online diary and calendar has been included. One can keep track of appointments and current tasks and maintain a "todoyet" file, all from a single desktop.

14) *Corpus Presenter Direct Viewer*
If for some reason one does not wish to use *Corpus Presenter*, one can still view texts, databases, images and listen to audio files with the current programme, which also allows for limited retrieval operations. The advantage here is one of speed, and of course one does not need to create a data set file to be able to view files present somewhere on disk.

15) *Corpus Presenter Database Manager*
The most important data format after texts is that of databases which arrange information in the form of a grid with rows and columns. There is greater inherent flexibility in databases, but they also require greater discipline in the collection of data and are most suited for large amounts of similarly structured data.

One application in the area of corpora is the processing of lexical material. The supplied database manager contains all the options needed for the collection, editing and export of formally structured data.

16) *Corpus Presenter Make Database*
To collect data with a database manager, one must create a database or use an existing one. Even in the latter case one may well find that one's conception of how data should be arranged alters with time, and so the need arises to change the structure of a database or just create a new one. In either case, the current utility will help to fulfil this task swiftly in an interactive, user-friendly environment.

17) *Corpus Presenter Quick Database*
For speedy processing of databases the present utility is useful. It has much less overhead than its parent programme *Corpus Presenter Database Manager*, and of course, does not show many of the functions of the latter. One feature deserving of attention here is the set of text macro options which saves on keying in repetitive text.

18) *Corpus Presenter Report Database*
When one wishes to export data from a database, it is necessary to specify how this is to be arranged in the output file generated. A small file called a report form determines how data from fields is arranged in the output text. One can have different report forms for one and the same database, which greatly increases flexibility. For instance, when outputting bibliographical data, one could use different report forms corresponding to different style sheets, which would obviate the necessity of hardwiring style-sheet preferences into the structure of the database. With the present utility one can design report forms interactively.

19) *Corpus Presenter Edit*
If one wishes to process corpus files, for instance, when one is collecting texts for a corpus, then one needs to use a so-called text editor, an editing utility which does not include formatting information in the files it creates. The present programme is intended for this, putting a whole range of options at one's disposal at the same time. Note that texts are tagged with this editor (see section 3 *Tagging texts* below). If the data for one's corpus does nonetheless necessitate the use of text formatting, eg for special fonts or word attributes, then one can

avail of the Rich Text Format mode and store text with formatting from within the current text editor and use these texts in a corpus which one processes with *Corpus Presenter* at a later stage.

20) *Corpus Presenter Word Processor*
The aim of a word processor is to allow the processing of formatted output, eg when preparing a text for printing. Hence the options it contains differ somewhat from a text editor. The supplied word processor has many formatting options concerning the appearance of a document which go beyond those of the text editor. The trade-off is a slight reduction of the speed of text processing.

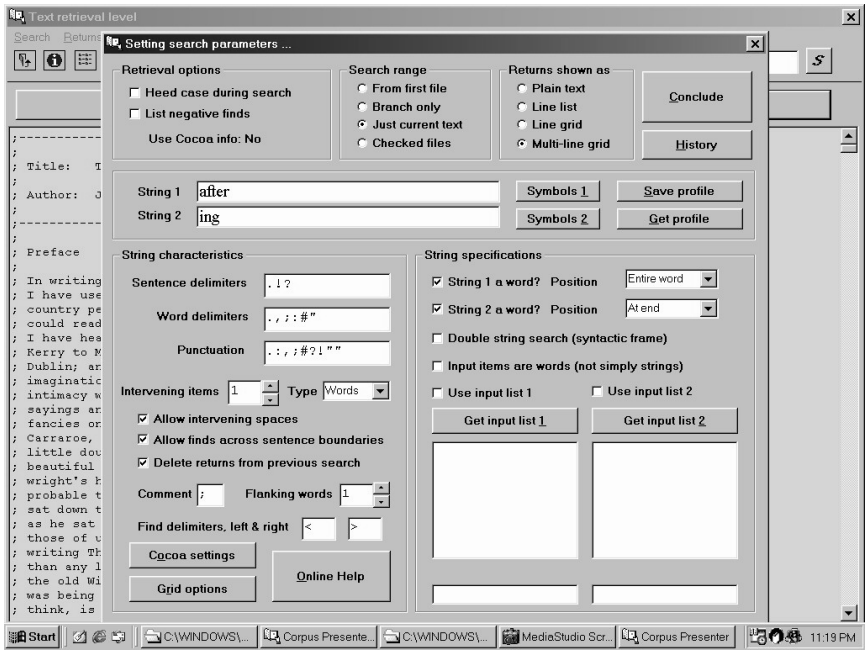
2 Using Corpus Presenter

The main purpose of *Corpus Presenter* is to display and interrogate an existing corpus, available either on a storage medium – such as a CD-ROM or disks – or available online. With online texts one must first of all download these to one's location as one cannot use local software, ie in one's computer, to search through data which is deposited somewhere on the internet. Nearly all files which one can download through the internet are in HTML (Hypertext Markup Language) format, one of the main formats which *Corpus Presenter* can handle. The second purpose of the current programme is to check on a corpus which one is compiling. Frequently linguists, alone or in groups, are engaged in gathering texts and arranging these as a corpus. This leads to the question of tagging.

Tagging in advance. With many corpora, tagging has been done in advance, ie the texts of a given corpus have been prepared in such a way as to render grammatical information in the corpus easily accessible to users. This is normally done by attaching grammatical tags to the word forms (to a selection, or in some rare cases to all words) of the texts. To utilise texts prepared in this way, one normally has to employ specially developed software, as is the case with the *International Corpus of English,* compiled at University College London. In the case of *Corpus Presenter,* there is no need to tag texts in advance, although one can do this if necessary (see section 3 below). The advantage is that one can begin to work with the bare texts of a corpus. Prior tagging of text does not preclude use with *Corpus Presenter.* In such an instance one may want to make use of the information accessible via the tagging. This is quite simple and can be done by entering tagging information on the retrieval level of *Corpus Presenter,* where users specify the information which is to be searched for in a range of texts from the corpus.

2.1 Retrieval tasks

There is a special level within *Corpus Presenter* dedicated to the retrieval of information from a corpus. It is here that one sets the various parameters for a search and where the information gleaned during such a search is returned and displayed to the user. As retrieval is such a central task, its operation is discussed in some detail in the present section, which hopefully will convey an impression of how the programme operates on this level.



The retrieval function allows one to locate virtually any string or strings in any of the texts of a corpus. For this to work properly, certain items of information and certain parameter settings are required. The most important of course is the search string itself, or strings, if one chooses to carry out a double string search (see below). When a search is performed, the string/strings entered is/are deposited in the history list which can be saved to disk and reloaded at a later point.

When searching for strings, *Corpus Presenter* can return the context in which it they occur. One can determine how much of this is shown by specifying

how many words to the right and left of the string are to be returned as well. As many corpora contain historical and/or foreign language texts, there is special provision for the use of non-standard characters. In addition, there are two phonetic (truetype) fonts,[2] supplied with the programme suite, which allows users to edit and print texts containing phonetic symbols from the International Phonetic Alphabet. It should also be mentioned here that a version of the Helsinki Corpus in which the Old and Middle English symbols (thorn, eth, ash) are displayed for what they are is available from the present author. This version has been linked to *Corpus Presenter* and can be used immediately without any further adaptation. Should the compilers of the Helsinki Corpus be agreeable to its distribution with *Corpus Presenter,* then this will be arranged.

In *Corpus Presenter* accurate retrieval is attained by paying attention to the following search parameters which determine the behaviour of the programme during the retrieval procedure.

1) *Case-sensitive*
If this parameter is not set, then uppercase and lowercase letters are treated in the same manner, that is, no distinction is made between capital and small letters. This also applies to any special symbols which can be used during a search.

2) *Double string search*
This type of search requires two strings, a first one which represents the left-hand section of a contextual frame and a second one which is the right-hand part. A typical example of a frame would be a phrase or part of a sentence. For instance, if one wishes to search for occurrences of *do* plus *have* in historical texts of English, say in the Helsinki Corpus, then one might enter the following.

FRAME SEARCH: Left *do* Right *have*

This would return finds like *do have*, *do certainly have*, etc. One can furthermore specify whether either or both strings are entire words or only a part of a word, as mentioned above.

3) *Allow across sentence boundaries*
A syntactic context for a frame search will more than likely be expected to occur within a sentence. If one wishes to deliberately search for a frame which straddles two sentences, then this can be specified as well. The set of delimiters for

sentences can be edited by the user. For instance, if one were dealing with Spanish texts, one would want to include the inverted exclamation and question mark symbols as possible sentence delimiters.

4) *Allow intervening spaces*
A frame search normally aims at returns consisting of several words, ie a phrase. However, it is equally possible to search for a word using a frame. For instance, if one wished to find all instances of negated adjectives in a text then one could enter a frame consisting of *un* and *able* and specify that intervening spaces are not allowed by removing the tick from the box for the current option. Such a search would return such tokens as *unacceptable, unbearable, unthinkable*, etc.

5) *String position in word*
This is a simple parameter which determines whether the units used for a search operation are entire words or only sections. The two latter possibilities here are *Beginning of word* and *End of word* respectively. For instance, if one wished to search for something like the perfective construction of Irish English, as in *She's after selling the car* 'She has just sold the car', one could enter *after* as String 1 and *ing* as String 2 and specify that the position of the latter is at the end of a word. This would ensure that in a sentence like *She's after bringing the dog* only the final *ing* is returned as a valid find for String 2.

On the other hand, one could choose the setting *Beginning of word* in a case like that discussed above under frame search. If one specified that *do* was only to be returned if found at the beginning of a word, then cases would be registered, like *don't,* which would allow for negated forms of *do* among the retrieval results.

6) *Intervening items*
The left and right of the frame can be separated by a specifiable number of intervening items (characters or words). If this is set to 0, then the left and right sections of the frame must be immediately adjacent. To allow simple adverbs in the above example, one would set the type of intervening item to *words* and the number to 1.

7) *Halt at string finds*
Setting this parameter will force *Corpus Presenter* to stop and display each find for the search string. If an automatic search is required, then this parameter is not set.

8) *Collect finds in list*
There is an internal array in *Corpus Presenter* which is filled with information about string finds. This list can be stored to disk or copied to the *Windows* clipboard via appropriate options.

9) *Range of search*
Because *Corpus Presenter* works with texts arranged as a layered tree, it is possible to specify the range of a search as 1) the current text, 2) those texts including and below the current node or 3) the entire data set.

10) *Save/get profile*
All the parameters specified for a certain search can be saved to disk and retrieved during another work session.

11) *Retrieval returns*
Finally, it should be mentioned that retrieval returns can be displayed in a grid or list. This information can be stored to disk as a table and later processed with the supplied table editor. The grid of returns can itself be edited in a number of ways. One can, for instance, select only some of the returns (those one regards as valid from the point of view of contents) and save these only. The returns can be arranged as columns, which can be sorted, selected, copied to clipboard or disk, etc.

COCOA PARAMETERS
One means of specifying various items of information about a corpus text is to mention these in a header at the beginning of each file. A system which is quite widespread among corpora is the *Cocoa* parameter set. This consists of up to 32 parameters with typical settings for certain file types. For instance, the texts of the Helsinki Corpus are all encoded with a *Cocoa* header, in which information is given about a following text. The settings can be used in *Corpus Presenter* to determine what files are examined during a retrieval operation.

3 Tagging texts

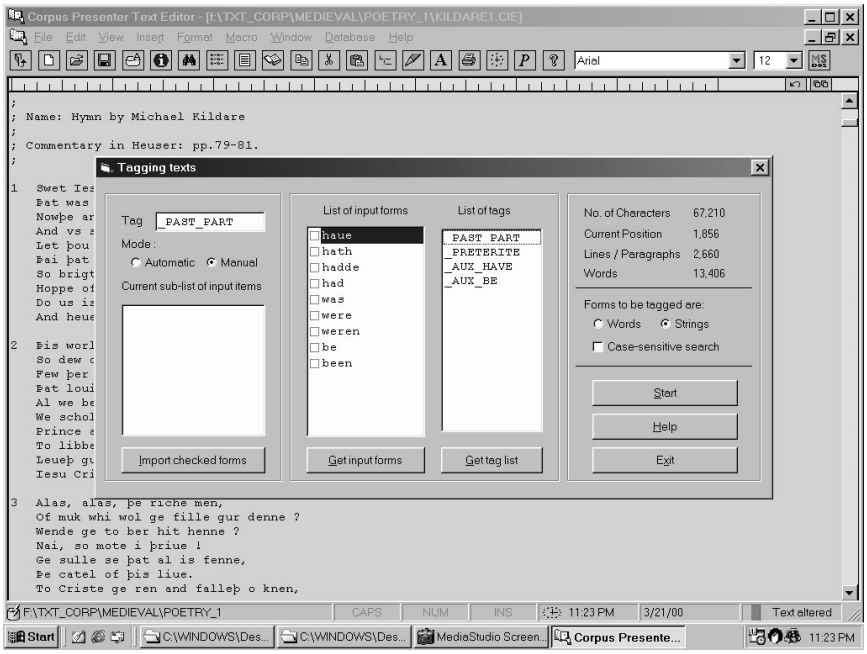
The text editor supplied with the current suite – *Corpus Presenter Edit* – has been designed as a flexible editing facility which can handle any number of files of any size which are stored either as plain ASCII texts or as Rich Text Format

files (the latter contain formatting information and can be read by virtually any word processor on nearly all operating system platforms). The only restrictions on the size and number of files is the amount of memory physically present in one's computer. On a computer with 64 MB of system memory, texts of several megabytes can be processed easily.

Tagging a text consists of attaching a grammatical label as a suffix to a word form. The user decides what category of label is to be suffixed to what word forms. Once this operation has been carried out, grammatical information can be retrieved from the texts of a corpus by referencing the tag suffixes. Very often the individual who does the tagging and the one who carries out the retrieval tasks are not the same. Note that the retrieval results using grammatical tags is only as good as the tagging is in the first place. In general one cannot reference semantic information in a corpus; ie a tagged corpus is primarily intended for retrieving morphological and possibly syntactic information. Before one starts tagging, one must copy one or more input forms into the list provided.

3.1 Preparing corpus texts

When preparing the texts for a corpus, one requires a text editor which does not contain too many formatting options. The reason for this is that, if the programme has several formatting possibilities, such as block justification, boxes, graphic image manipulation, object embedding and the like, then the speed of the programme is slowed down considerably and the upper limit on file size is reduced. This is where a quick text editor is useful. The present programme will process plain ASCII texts without any special alterations to the texts or any instructions on how to save them to disk (contrast this with commercial word processors). If one wishes to have more formatting commands at one's disposal, then one can avail of *Corpus Presenter Word Processor,* which offers a much wider range of word processing options both for text processing and printing.



List of input forms
This is the basis for the tagging operation. It consists of a list of forms determined by the user in advance. One can create a list with *Corpus Presenter Edit* itself and store this to disk. Such a list consists of a number of lines, each with just a single form on it.

List of tags
This list is formally similar to the previous one, with the difference that it contains the tags which one may wish to use for a tagging operation. With a further option one can load a file and use it as the current tag list. The maximum number of tags and of input forms is 512 items in each case.

For any run of the tagging function one must select a single tag and have chosen at least one item from the list of input forms. One can select a word from the input forms by clicking on the check box beside it and then select the option *Import checked forms*. These forms are now entered in the sub-list, and with a

tag in the box above one is ready to begin the tagging operation. Attention should be paid in this connection to the various parameters for tagging as indicated below.

Tagging parameters

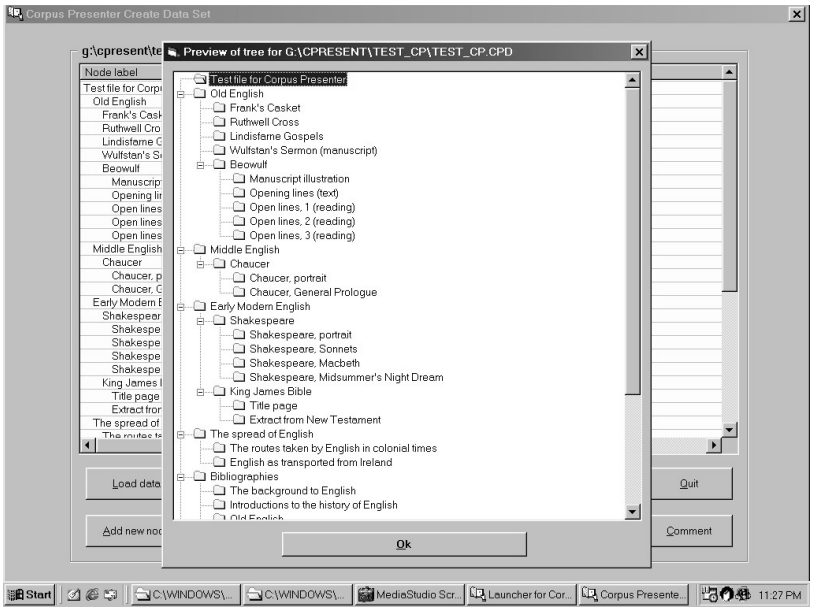
1. *Words or strings.* Specifies if only words - or any string – can be tagged.
2. *Case-sensitive search.* Determines whether small and capital letters are distinguished.
3. *Automatic or manual.* Here you can decide whether *Corpus Presenter Edit* halts at each find and asks the user to confirm whether a form is to be tagged or not. Note that, with manual tagging, you can also edit the finds in the current text as you proceed.

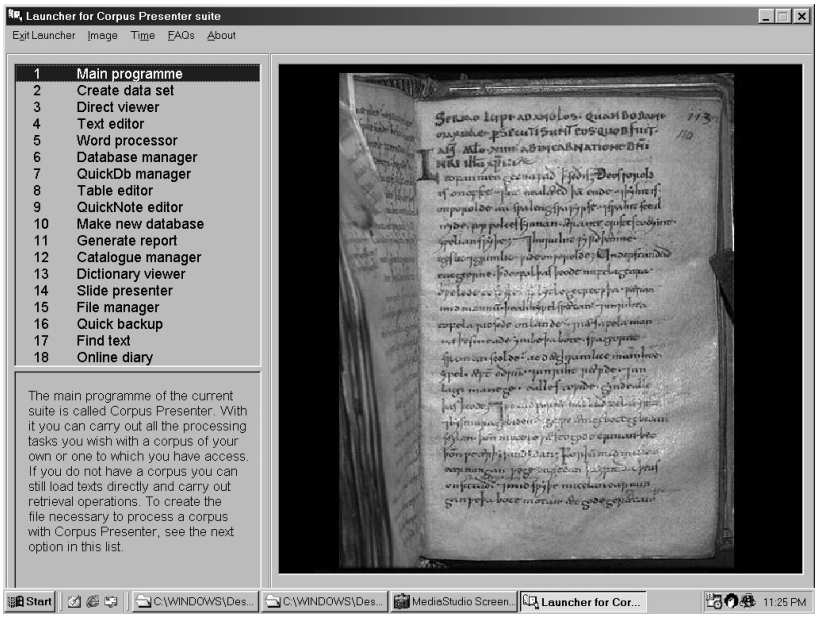
4 Linking a corpus to Corpus Presenter

Linking a corpus – one's own or one you have acquired from another source – does not entail altering the corpus in any way. All that is necessary is that a single file be created which will control the display of the corpus under *Corpus Presenter* and control what files are used for any retrieval tasks carried out. The present author has already produced the necessary control files for the Helsinki Corpus, for the Corpus of Early English Correspondence and the Corpus of Older Scots[3] without manipulating the corpus files and hence without infringing on the copyright of the compilers. The control file necessary to make an existing corpus accessible to *Corpus Presenter* can be created interactively by availing of the supplied programme *Corpus Presenter Create* (see above).

This control file is technically referred to as a *data set* file and contains settings for various parameters, which determine the appearance and operation of *Corpus Presenter* along with a list of the files which form part of the corpus in question. A data set file – called `TEST_CP.CPD` – has been included with *Corpus Presenter* and allows you to see what kinds of file can be included in a corpus and to test the different functions of the programme. However, if one wishes to design one's own data, set one can do so by either creating a new data set file or adapting the supplied one to suit one's needs. The programme *Corpus Presenter Create* enables one to alter all the parameters of a data set file and to specify what files are to be displayed using this data set with *Corpus Presenter.*[4] There is also a preview function with which one can see the tree display of the files included in one's data set without having to load *Corpus Presenter.*

The current programme can be started directly from *Corpus Presenter* or via the supplied launcher programme (see above).


Corpus Presenter Create Data Set
Preview of tree for G:\CPRESENT\TEST_CP\TEST_CP.CPD
Test file for Corpus Presenter
Old English
Frank's Casket
Ruthwell Cross
Lindisfarne Gospels
Wulfstan's Sermon (manuscript)
Beowulf
Manuscript illustration
Opening lines (text)
Open lines, 1 (reading)
Open lines, 2 (reading)
Open lines, 3 (reading)
Middle English
Chaucer
Chaucer, portrait
Chaucer, General Prologue
Early Modern English
Shakespeare
Shakespeare, portrait
Shakespeare, Sonnets
Shakespeare, Macbeth
Shakespeare, Midsummer's Night Dream
King James Bible
Title page
Extract from New Testament
The spread of English
The routes taken by English in coloniel times
English as transported from Ireland
Bibliographies
The background to English
Introductions to the history of English
Ok
Load data
Add new no
Quit
Comment
Start
C:\WINDOWS\...
C:\WINDOWS\...
MediaStudio Scr...
Launcher for Cor...
Corpus Presente...
11:27 PM


Launcher for Corpus Presenter suite
Exit Launcher
Image
Time
FAQs
About
1 Main programme
2 Create data set
3 Direct viewer
4 Text editor
5 Word processor
6 Database manager
7 QuickDb manager
8 Table editor
9 QuickNote editor
10 Make new database
11 Generate report
12 Catalogue manager
13 Dictionary viewer
14 Slide presenter
15 File manager
16 Quick backup
17 Find text
18 Online diary
The main programme of the current suite is called Corpus Presenter. With it you can carry out all the processing tasks you wish with a corpus of your own or one to which you have access. If you do not have a corpus you can still load texts directly and carry out retrieval operations. To create the file necessary to process a corpus with Corpus Presenter, see the next option in this list.
Start
C:\WINDOWS\Des...
C:\WINDOWS\Des...
MediaStudio Screen...
Launcher for Cor...
11:25 PM

4.1 Structure of a data set file

A data set file contains all the information needed for displaying the files of a corpus correctly in tree form. For each node of a tree three pieces of information are specified. In addition there are eleven parameters, which are set at the beginning of the file and which determine the location of the corpus files and the manner in which they are displayed, along with the names of 1) the manual file for a corpus, 2) a 'Frequently Asked Questions' (FAQ) file and 3) a 'Fact Sheet' file.

The information for the nodes of the tree are arranged as follows. The first is the description to be used as a label for a node (plain text). The second is the file associated with this node. If one enters `DUMMY.RTF` here, then no file is displayed. This is necessary because there will be nodes in a tree which are empty; ie they are just links to other nodes further down the tree. Indeed it is normal, though not essential, that only the terminal nodes of a tree contain actual file references. The third item of information usually consists of three asterisks. The reason it is there at all is that, with audio files, you may wish to display an image file in the background. If you now specify an audio file (with the extension `WAV`) as item no. 2 and an image file as item no. 3, then the latter will be displayed while the former is played. By these means you could, for example, display a map of a region and play an audio file with the speech of that area at the same time. An example of this function is to be found in the sample data set supplied with *Corpus Presenter.*

You will notice that the description of many nodes is indented and represents the means by which one specifies what level in a tree the node is to be displayed on. The principle is as follows: every four spaces at the beginning of a label represent an indent of one level below the first, ie no spaces indicate a node on the top-most level (level 1), four spaces indicate that the node is on level 2, eight spaces on level 3, twelve on level 4, sixteen on level 5 and twenty on level 6. A maximum of 6 levels is permissible.

Sample section of data set file for the Helsinki Corpus *(beginning, early Old English)*

```
Old English
DUMMY.RTF
***
    I   ( - 850)
DUMMY.RTF
***
        Documents
DUMMY.RTF
***
            Documents 1 (Harmer, Robertson, Birch)
```

```
CODOCU1
***
        Undefined text type (verse)
DUMMY.RTF
***
            Caedmon's Hymn; Bede's Death Song; The Ruth-
well Cross; The Leiden Riddle
CONORTHU
***
    II  (850-950)
DUMMY.RTF
***
        Law
DUMMY.RTF
***
           Alfred's Introduction to Laws, Laws (Alfred),
Laws (Ine)
COLAW2
***
etc
```

5 Availability of Corpus Presenter

The programme suite *Corpus Presenter* comes on a single CD-ROM, from which it can be installed onto any computer running under Microsoft *Windows 95* (or higher) and with at least 30 MB of free space and at least 32 MB of system memory. It will also run on recent versions of the Apple Macintosh which can run *Windows* in a so-called emulation mode. The question of general availability is still a subject of discussion between the author and possible distributors of the software. It is envisaged that a decision on this will be reached shortly. Scholars interested in obtaining this software should log onto the following homepage: `http://www.uni-essen.de/anglistik` and then click on the button "Projects and Activities" on the left-hand side of the screen. There is an entry "Corpus Presenter", where information on the availability of the software will be announced as soon as possible.

There is a manual accompanying the *Corpus Presenter* suite. It is approximately 180 pages long and contains much information on how to gain maximum benefit from the use of the package. Hopefully, the final distribution form will be a combination of manual and CD-ROM.

Notes

1. For reasons of space, only a brief indication of the functions which the various programmes embody can be given in this article. Each programme has a

comprehensive online help, and there is a manual of some 180 pages accompanying the software.

2. These replace to a certain extent those supplied for DOS with the programme suite *Lingua Font* (see Hickey 1993c).
3. The latter two corpora have also been compiled at the English Department of Helsinki University; see Nevalainen (1997), Raumolin-Brunberg (1997) and Meurman-Solin (1997) for representative discussions of these corpora and their aims. For the main Helsinki Corpus, see the exhaustive description in Kytö (1993). Mention should also be made of the ongoing work of Irma Taavitsainen and her colleagues on an historical corpus of medical texts, also at Helsinki University.
4. Data set files are plain ASCII texts and can be processed using any text editor, such as the supplied one *Corpus Presenter Edit*. This file should not be saved in RTF format (or that of any commerical word processor), as it would then no longer function properly as a data set file.

References

Hickey, Raymond. 1993a. *Lexa. Corpus processing software, 3 Vols. Vol.1: Lexical analysis. Vol.2: Database and corpus management. Vol.3: Utility library.* Bergen: Norwegian Computing Centre for the Humanities.

Hickey, Raymond. 1993b. Corpus data processing with Lexa. *ICAME Journal* 17: 73–96.

Hickey, Raymond. 1993c. *LinguaFont. Language fonts and design software*. Bergen: Norwegian Computing Centre for the Humanities.

Hickey, Raymond, Merja Kytö, Ian Lancashire and Matti Rissanen (eds). 1997. *Tracing the trail of time. Proceedings from the Second Diachronic Corpora Workshop, Toronto, May 1995*. Amsterdam – Atlanta, GA: Rodopi.

Kytö, Merja. 1993. *Manual to the diachronic part of the Helsinki Corpus of English Texts*. 2nd edition. Helsinki: Department of English, University of Helsinki.

Meurman-Solin, Anneli. 1997. Text profiles in the study of language variation and change, in Hickey et al, 199–214.

Nevalainen, Terttu. 1997. Ongoing work on the *Corpus of Early English Correspondence*, in Hickey et al, 81–90.

Raumolin-Brunberg, Helena. 1997. Incorporating sociolinguistic information into a diachronic corpus of English, in Hickey et al, 105–118.